LIBRARY OF THE
UNIVERSITY OF ILLINOIS
AT URBANA-CHAMPAIGN

510.84
I&63c
no.61-70

AUG 5 1976
The person charging this material is responsible for its return to the library from which it was withdrawn on or before the **Latest Date** stamped below.

**Theft, mutilation, and underlining of books are reasons for disciplinary action and may result in dismissal from the University.**

UNIVERSITY OF ILLINOIS LIBRARY AT URBANA-CHAMPAIGN

ENGINEERING

CONFERENCE ROOM

INTERLIBRARY LOAN

JAN. 1 8 REC'D

APR 8 1980

MAR 1 8 REC'D

PHOTO REPRODUCTION

OCT 22 REC'D

PHOTO REPRODUCTION

NOV 1 6 REC'D

PHOTO REPRODUCTION

OCT 0 1 REC'D

PHOTO REPRODUCTION

OCT 2 7 REC'D

PHOTO REPRODUCTION

PHOTO REPRODUCTION

OCT 2 3 REC'D

L161—O-1096

Digitized by the Internet Archive
in 2012 with funding from
University of Illinois Urbana-Champaign

http://archive.org/details/useofilliacivtos67cich

510.84
I263c
no.67

Engin.

CONFERENCE ROOM

ENGINEERING LIBRARY
UNIVERSITY OF ILLINOIS
URBANA, ILLINOIS

Center for Advanced Computation

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
URBANA, ILLINOIS 61801

CAC Document No. 67

# THE USE OF ILLIAC IV TO SOLVE NUMERICAL FLUID DYNAMICS PROBLEMS

By

Paul T. Cichy

February 27, 1973

CAC Document No. 67

# THE USE OF ILLIAC IV TO SOLVE NUMERICAL FLUID DYNAMICS PROBLEMS

## A STATUS REPORT

by

Paul T. Cichy

Center for Advanced Computation
University of Illionis at Urbana-Champaign
Urbana, Illinois 61801

February 27, 1973

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and was monitored by the U. S. Army Research Office-Durham under Contract No. DAHC04-72-C-0001, and by the National Science Foundation (NSF GK 2813X).

LIBRARY

ABSTRACT

The usefulness of ILLIAC IV in solving numerical fluid dynamics problems is studied by developing a parallel algorithm for the unsteady two dimensional flow of an incompressible fluid around a circular cylinder. Two cases are studied, the symmetric case, for which an operational GLYPNIR program is included, and the asymmetric case, for which a partially completed GLYPNIR program is attached. The programming of a parallel algorithm for ILLIAC IV is more difficult than programming a comparable serial algorithm but this additional difficulty is compensated for by the greatly reduced computation time required to solve the problem.

TABLE OF CONTENTS


| | | Page |
|---|---|---|
| 1. | INTRODUCTION | 1 |
| 2. | THE ILLIAC IV SYSTEM | 2 |
| | 2.1 The ARPA Network | 3 |
| | 2.2 The Ames Site | 3 |
| | 2.3 The ILLIAC Array | 6 |
| | 2.4 Programming Languages | 8 |
| 3. | THE CHOICE OF A PROBLEM | 9 |
| 4. | THE SYMMETRIC FLOW AROUND A CYLINDER | 11 |
| | 4.1 The Problem | 11 |
| | 4.2 Development of the Algorithm | 12 |
| | 4.3 Programming the Algorithm | 24 |
| | 4.4 Testing the Algorithm | 24 |
| | 4.5 Problems with this Symmetric Formulation | 26 |
| | 4.6 Conclusions | 27 |
| 5. | THE ASYMMETRIC FLOW AROUND A CYLINDER | 28 |
| | 5.1 Description of the Problem | 28 |
| | 5.2 Development of the Algorithm | 33 |
| | 5.3 Programming the Algorithm | 39 |
| | 5.4 Current Status of Program | 48 |
| | 5.5 Conclusions | 48 |
| 6. | FUTURE PROGRAMS IN THIS PROJECT | 49 |
| | REFERENCES | 50 |
| | APPENDIX A. CICHY/GLYSOND | 51 |
| | APPENDIX B. CICHY/GLYDIRECT | 64 |
| | APPENDIX C. CICHY/MASTER | 82 |
| | APPENDIX D. CICHY/DIRECT | 105 |
| | APPENDIX E. CICHY/UNS | 134 |

## 1. INTRODUCTION

The numerical solution of a large class of fluid dynamic problems is currently impractical due to the enormous amounts of computation time needed to generate useful results on presently available serial processing computers. One approach to increasing computation speed using established hardware is to construct a computer which interprets a single instruction stream but causes arithmetic and logic operations to be performed on many sets of data simultaneously. This type of computer architecture has been given the descriptive terms "parallel processor" or "array processor".

ILLIAC IV is a new parallel processing computer designed by the University of Illinois and Burroughs Corporation for the Advanced Research Projects Agency (ARPA). ILLIAC IV interprets a single instruction stream in its central control unit (CU) and then causes operations to take place in 64 separate processing elements (PE), each PE acting on its own set of data. The hardware was constructed by Burroughs at Paoli, Pennsylvania, and installed at the Ames Research Center, Moffet Field, California. The computer was accepted by ARPA in December 1972 and is expected to be operational by June 1973. The Ames facilities are accessable to a large group of users through the ARPA network, a national network connecting many Universities, Government Laboratories, and Government Contractors.

There are several research areas where a very fast computer is currently needed in order to solve problems requiring the processing of large amounts of data. Problems in numerical fluid dynamics, studies in weather prediction, analysis of seismographic information to detect nuclear explosions, and the study of artificial intellegence all represent areas of active interest.

The project discussed in this report is directed toward the solution of numerical fluid dynamics problems. The overall objectives are:

1) to determine what difficulties may be encountered in programming a parallel processing computer and to suggest ways to minimize them,
2) to develop a parallel algorithm which will solve a problem previously solved by serial techniques and then compare the two results for ease of programming, speed of obtaining the solution, and accuracy, and, finally,
3) to draw some conclusions about the general usefulness of ILLIAC IV in solving numerical fluid dynamic problems.

Some of these objectives cannot be fully attained until ILLIAC IV is operational. This report presents the progress made toward meeting these objectives from June 1972 to February 1973. The initial section describes the ILLIAC IV architecture as it exists today. The following sections discuss the choice of a numerical fluid dynamic problem and the preparation of two algorithms to solve that problem. These sections are followed by several conclusions on the use of ILLIAC IV based on the experience gained during this project.

The project is jointly sponsored by the Department of Chemical Engineering (Dr. T. J. Hanratty) and the Center for Advanced Computation (Dr. A. Sameh).

## 2. THE ILLIAC IV SYSTEM

The ILLIAC IV array processor does not exist as a totally separate entity but is intimately linked with peripheralequipment which manages its activity and performs the services it requests. The user approaches

the ILLIAC IV system through the ARPA network and places a request for computing time with the TENEX managing system. TENEX in turn schedules and runs the job on the ILLIAC IV array processor and fulfills all the array's INPUT/OUTPUT (I/O) requests. A brief description of each section of the ILLIAC IV system follows. A more complete description has been prepared by S. A. Denenberg [1].

## 2.1 The ARPA Network

The ARPA network is designed to make available to a large number of users a wide variety of computational resources. Some of the locations attached to the network are indicated in the sketch in Figure 1. Network users have available such large computers as IBM 360's, a Burrough's B6700, and several PDP 10's and PDP 11's. Each location is served by an Interface Message Processor (IMP) which controls the flow of information over the net. At present, the net is functional, but due to local conventions at each site, much care is needed in its use. A network protocal now under development should make its use much easier.

## 2.2 The Ames Site

The Ames location is managed by a PDP 10 using a TENEX system as illustrated in Figure 2. Network users communicate with the TEN through a special control language. Documentation on this language is being prepared and should be available soon. The TENEX managing system controls a large disk data storage area, a UNICON laser storage device with a trillion bits of data storage and a B6700 for compiling programs for ILLIAC IV. It also manages the use of the ILLIAC IV array and handles all requests for I/O to and from the array. The TENEX managing system is being brought up slowly. At present it is capable of storing programs in its disk file

ARPA NETWORK

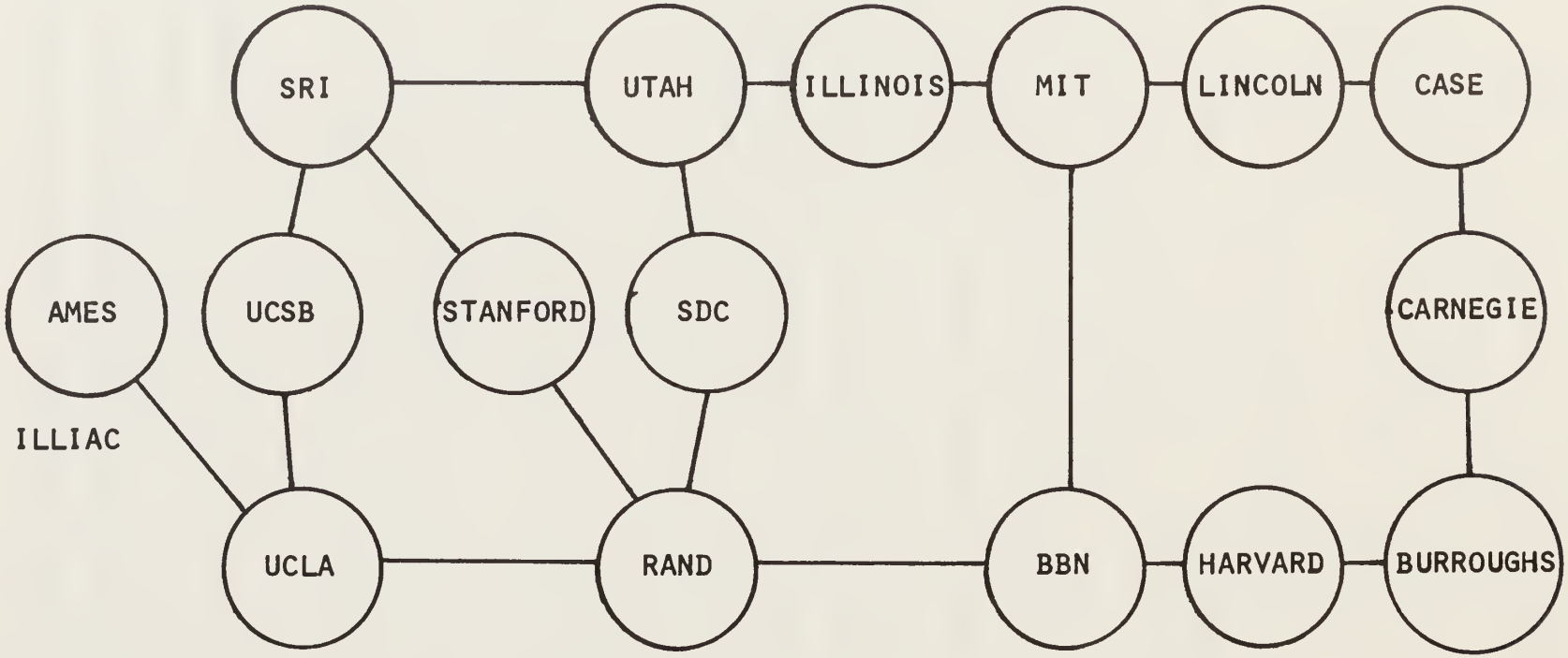



FIGURE 1

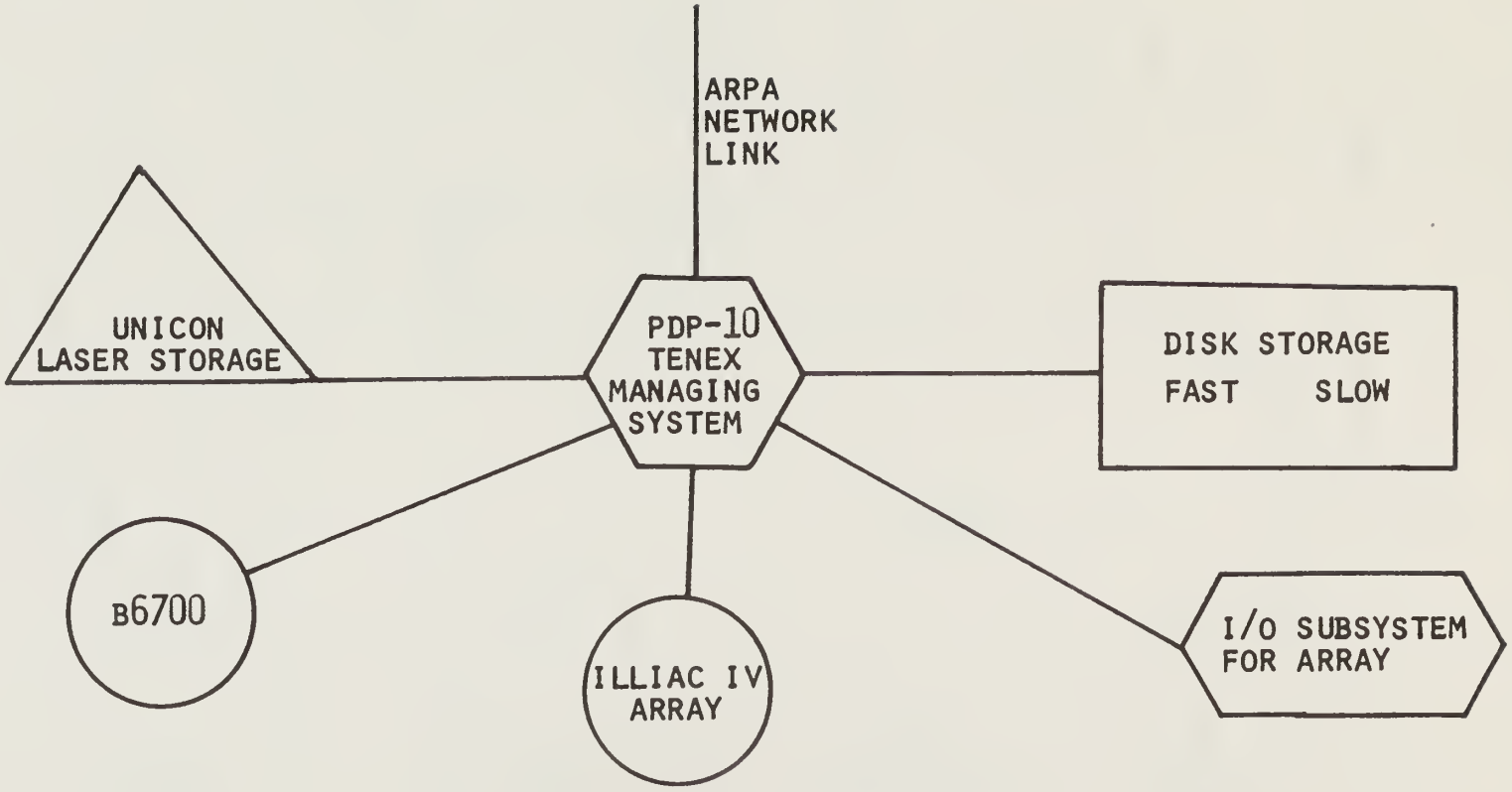

ARPA
NETWORK
LINK
UNICON
LASER STORAGE
PDP-10
TENEX
MANAGING
SYSTEM
DISK STORAGE
FAST SLOW
B6700
ILLIAC IV
ARRAY
I/O SUBSYSTEM
FOR ARRAY

FIGURE 2

system and compiling ASK assembly language programs, but it is not able to run jobs on the ILLIAC array. Current projections are that it will be functional in April. Since all access to the ILLIAC array will be over the ARPA net and through the TENEX managing system, any use of the machine must wait until TENEX is functional.

## 2.3 The ILLIAC Array

The architecture of ILLIAC IV is unique, and an understanding of this architecture is very important to those who would program ILLIAC efficiently. The computer is composed of two major sections as shown in Figure 3. The central control unit (CU) and the processing elements (PE), 64 units each having an arithmetic and logic unit (ALU) and a process element memory (PEM). The instruction stream is interpreted by the CU. The CU then sends messages to the processing elements to conduct the indicated data handling and arithmetic operations. Each PE executes the same command but operates on the specific data stored in its own processing element memory. Thus, when repetitive operations are to be performed on different but similar data sets, ILLIAC may be used to greatly increase the speed of computation by operating on 64 data sets at a time. It is projected that ILLIAC will be capable of executing 200 million instructions per second.

One of the major limitations of the ILLIAC array is the small size of each PEM. The maximum capacity of each PEM is 2048 words, that is approximately a 2K memory in each processing unit. For many problems this means that the storage cannot be core contained and that time consuming exchanges of data between the array and peripheral storage units must be carried out.

ILLIAC operates with a 64 bit word allowing 48 bits of mantissa and 15 bits of exponent. This large word size will allow increased accuracy

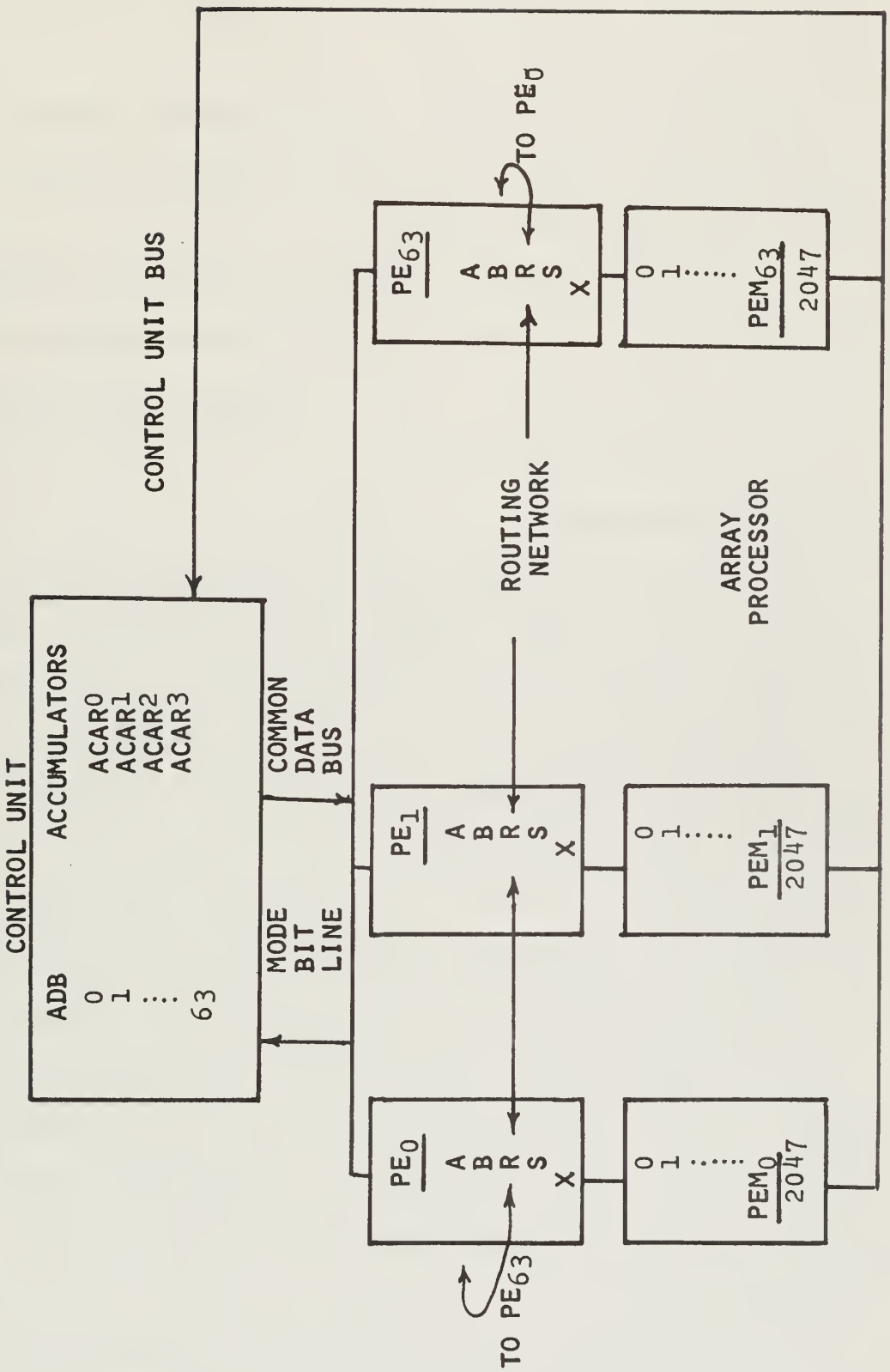

CONTROL UNIT BUS
CONTROL UNIT
ADB
0
1
...
63
ACCUMULATORS
ACAR0
ACAR1
ACAR2
ACAR3
MODE BIT LINE
COMMON DATA BUS
ROUTING NETWORK
ARRAY PROCESSOR
PE0
PE1
PE63
A
B
R
S
X
PEM0
PEM1
PEM63
2047
TO PE63
TO PE0

FIGURE 3

over most machines using single precision arithmetic. Double precision arithmetic is not currently available on ILLIAC IV.

Information can be transferred from one processing element to another through the routing network. This feature is especially important when solving partial differential equations.

Detailed information on the operation of ILLIAC IV is presented in The ILLIAC IV Systems Characteristics and Programming Manual published by The Burroughs Corporation. A copy of the manual may be obtained through the Center for Advanced Computation.

## 2.4 Programming Languages

The unusual architecture of ILLIAC IV required the development of an entirely new type of Assembly language, one which would allow the user to fully exploit the features of this parallel machine. Assembly Language K (ASK) was developed for this purpose. Experience quickly demonstrated the need for a higher level language and in response to that need, GLYPNIR was developed. Because ASK and GLYPNIR are tied closely to the specific architecture of ILLIAC IV, it is particularly important to understand this architecture fairly well before attempting to learn either language. Also, given the current status of the literature describing GLYPNIR, it is important to have a fairly good understanding of Algol before attempting to learn GLYPNIR. The language is relatively easy to use for a programmer with a knowledge of the ILLIAC IV architecture. A GLYPNIR Manual has been prepared by Terry Layman and David Baer and information on how to obtain a copy can be obtained from CAC.

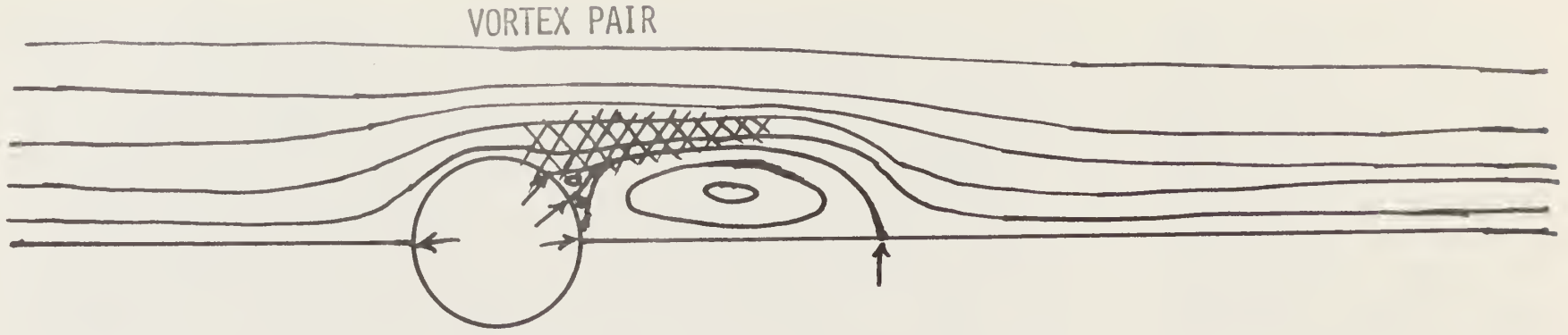
The GLYPNIR compiler produces ASK code which then must be compiled into machine language by the ASK compiler. Examination of the ASK code

produced by GLYPNIR indicates that the price paid for the convenience of a higher level language is an ASK code which is slightly inefficient. This means that computation time can sometimes be decreased significantly if often used loops in the program are coded in ASK. As a general strategy, an algorithm should be first coded in GLYPNIR. The ASK generated for internal loops should then be examined for inefficiencies. Efficient ASK code can then be substituted in the GLYPNIR program through the use of CODE statements.

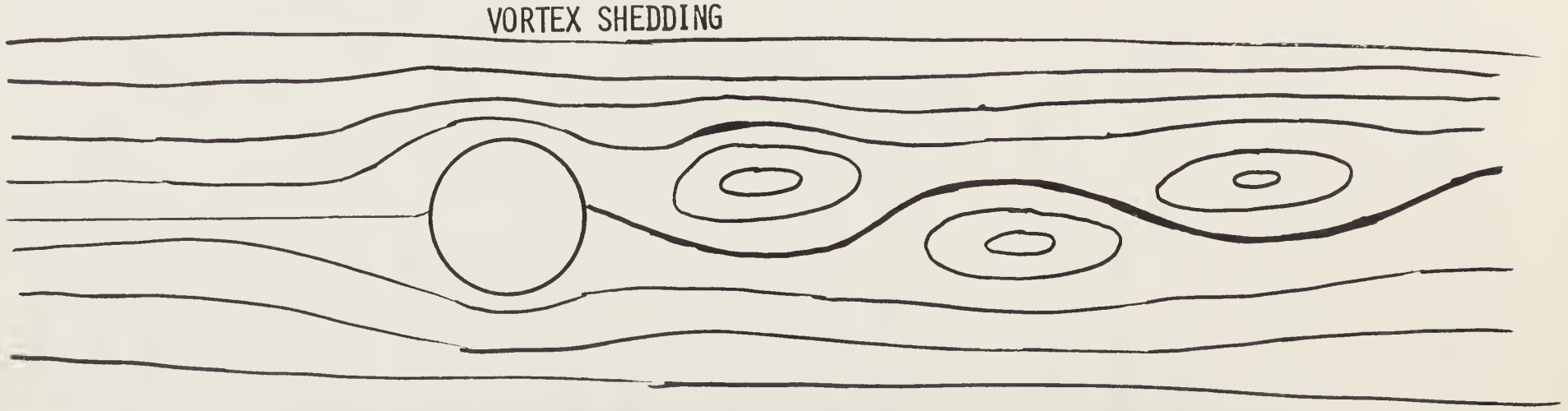
## 3. THE CHOICE OF A PROBLEM

One objective of this project was to develop a parallel algorithm to solve a numerical fluid dynamics problem on ILLIAC IV. There were several criteria which an appropriate problem had to meet. The problem must have been solved previously by serial techniques and the results of this solution must be available. The initial problem must be of a size and complexity which would allow it to be solved on ILLIAC IV under the uncertain conditions prevailing during the start up of a new computer. It was also desirable that the problem illustrate some interesting fluid dynamic phenomena.

The unsteady flow of an incompressible fluid around a circular cylinder was choosen as the problem to be solved. Two cases were considered, symmetric flow and asymmetric flow, and representative flow patterns for each case are illustrated in Figure 4. The symmetric flow case had been studied by J. S. Son [4] at the University of Illinois and his numerical results were on hand. This was an ideal initial problem because it could be core contained and required no special treatment of the boundary conditions. Furthermore, it was an interesting fluid dynamic problem in that it contained a growing boundary layer with separation and the formation of primary and secondary vorticies.




SYMMETRIC CASE




UNSYMMETRIC CASE

FIGURE 4

The asymmetric case had been treated by D. C. Thoman and A. A. Szewczyk [5] and the results were available in their report. This problem was more difficult requiring memory swaps and special treatment of the boundaries and, therefore, was attacked only after the symmetric algorithm was completed. However, the asymmetric case, where vortex shedding occurs, is more realistic physically when considering flows with Reynolds numbers in excess of 40.

The development of the algorithm for the symmetric case is presented in the next section. The asymmetric case is discussed in a later section.

## 4. THE SYMMETRIC FLOW AROUND A CYLINDER

### 4.1 The Problem

The unsteady two dimensional symmetric flow of an incompressible fluid around a circular cylinder was choosen as the first problem to be studied. This problem was investigated previously by J. S. Son [4] and his theoretical development and boundary conditions were adopted directly.

The governing equations were developed from the two dimensional Navier-Stokes equations. These equations were differentiated in such a way that they could be combined to eliminate the pressure terms. The resulting equation along with the continuity equation formed a pair of coupled nonlinear equations describing the fluid flow. These equations were converted to a stream function and vorticity formulation and non-dimensionalized. The independent variables were then transformed to adjust the coordinate spacing so that the boundary layer region would be adequately covered with mesh points.

The flow was assumed symmetric, thus the transverse velocity and the vorticity were assigned a value of zero along the centerline of

the flow. The fluid velocity at the outer edge of the flow field was taken to be parallel to the incoming flow and the velocity through the cylinder surface was taken as zero. The surface vorticity was estimated from the stream function distribution near the surface by expanding the vorticity function in terms of the stream function in the vicinity of the surface using the definition of vorticity. The pertinent equations developed in Son's thesis and the appropriate boundary conditions are given in Figure 5.

## 4.2 Development of the Algorithm

### 4.2.1 Introduction

The parallel algorithm developed to solve this problem was patterned after the serial algorithm used by Son. The general solution scheme is shown in Figure 6. The flow is assumed to start impulsively from rest and the initial stream function values are taken as those for the potential flow case. The finite difference form of the stream function equation is then used to adjust these values to be compatable with the boundary conditions. The initial vorticity in the flow field is assigned a value of zero. The surface vorticity at time zero is determined from the stream function distribution.

When all the initial values have been determined, dimensionless time is advanced by one increment and a finite difference form of the vorticity transport equation is used to calculate vorticities at the new time level. The stream function values are then advanced by a finite difference form of the stream function equation, using the latest values of vorticity. Two different techniques for updating the stream function were used, an iterative method and a direct method. These will be discussed in more detail in a later section.

VORTICITY TRANSPORT EQUATION

$$E^2 \frac{\partial \zeta}{\partial \tau} - \frac{\partial \psi}{\partial \eta}\frac{\partial \zeta}{\partial \xi} + \frac{\partial \psi}{\partial \xi}\frac{\partial \zeta}{\partial \eta} = \frac{2}{N_{Re}}\left[\frac{\partial^2 \zeta}{\partial \xi^2} + \frac{\partial^2 \zeta}{\partial \eta^2}\right]$$

STREAM FUNCTION EQUATION

$$E^2 \zeta = \frac{\partial^2 \psi}{\partial \xi^2} + \frac{\partial^2 \psi}{\partial \eta^2}$$

INITIAL CONDITION FOR STREAM FUNCTION

$$\psi = e^{\pi \xi}\left[1 - e^{-2\pi \xi}\right] \sin(\pi \eta)$$

BOUNDARY CONDITIONS

$\xi = 0$

$\psi = 0 \; ; \; \frac{\partial \psi}{\partial \xi} = 0$

$\zeta = E^{-2}\frac{\partial^2 \psi}{\partial \xi^2}\Big|_{\xi=0}$

$\xi \to \infty$

$\psi \to e^{\pi \xi} \sin(\pi \eta)$

$\zeta = 0$

$\eta = 0$ AND $1$

$\psi = 0$

$\zeta = 0$

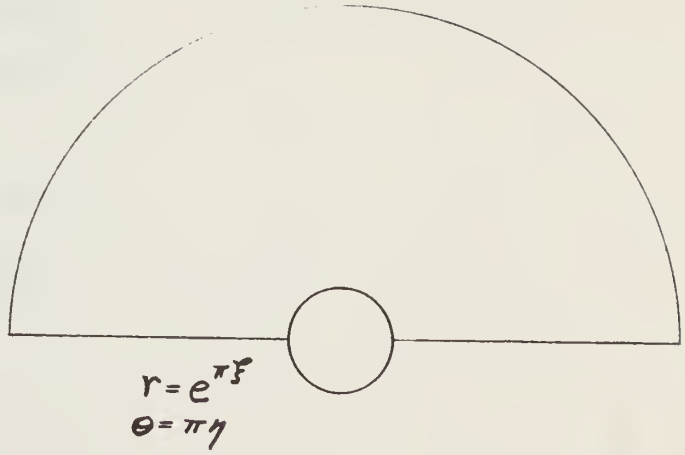



FIGURE 5

COMPUTATIONAL ALGORITHM

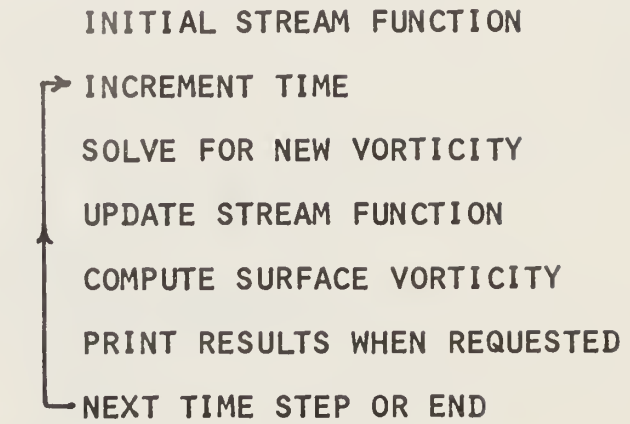



FIGURE 6

After the vorticity and stream function values have been updated in the interior of the flow region, the surface vorticity is adjusted to be compatible with the current stream function distribution near the surface. If requested, the value of all variables at this time level can be printed out at this point. The computation can then proceed by incrementing time and repeating the steps described above or it may be terminated.

The following sections treat in more detail each of the steps in the basic algorithm.

### 4.2.2 The initial solution

The values of the stream function at time zero were obtained from the analytic equations for the potential flow of a fluid of infinite extent about a circular cylinder. Since the fluid is assumed to be set into motion impulsively, at time zero, the boundary layer will be confined to a vanishingly thin region next to the surface of the cylinder and will not affect the rest of the flow field. Since viscous effects are confined to the boundary layer, the motion of the majority of the flow field is inviscid and the potential flow solution describes the motion well. Due to the need to deal with a boundary which is not at infinity but at some large but finite distance from the cylinder, a slight adjustment must be made in the analytic infinite extent values of the stream function. This is accomplished with the use of the finite difference form of the stream function equation as described below.

The actual surface vorticity at time zero should be infinite since, as the fluid impulsively assumes a velocity, a momentary velocity discontinuity exists at the surface. However, in order to obtain a tractable numerical solution, a large but finite vorticity is calculated from the stream function distribution.

The accuracy of the numerical solution at short times is in question due to the inability of the numerical scheme to account rigorously for the infinite surface vorticity and due to the difficulty in resolving the very thin boundary layer during these first few time steps. The effect that this initial error might cause in later calculations has not been determined but it is expected that it is confined to short times.

One method to improve the accuracy of the short time solution would be to use an analytic representation of a growing boundary layer to determine the values of the stream function and vorticity near the cylinder surface at a time when the boundary layer was still small with respect to the cylinder radius but when it was large enough to be resolved by the numerical scheme. The surface vorticity at this time would be finite.

4.2.3 The vorticity transport equation

The Alternating Direction Implicit (ADI) method was used to obtain the values of vorticity at the next time level. This method has been shown to be stable for fairly large time increments. Two steps are required to reach the next time level. The finite difference equations for each step are shown in Figure 7. During the first half time step the solution is taken to be implicit in the rows and explicit in the columns. In the next half time step the solution is implicit in the columns and explicit in the rows. This formulation results in the need to solve a tridiagonal matrix to obtain the implicit vorticity values in each row or column as the case may be. The Thomas Method represents a particularly simple method of solving these tridiagonal matricies. The solution is obtained by a series of calculations involving sequential terms in the coefficient matrix.

The full power of ILLIAC IV is realized only when all 64 PE's are active 100% of the time. Although this is seldom accomplished, maximum

## NUMERICAL SOLUTION OF THE VORTICITY TRANSPORT EQUATION

### ADI METHOD (ALTERNATING DIRECTION IMPLICIT)

STEP 1

$$f(\psi^n)\,\zeta_{i-1,j}^{n+1/2} + g(\zeta)\,\zeta_{i,j}^{n+1/2} + h(\psi^n)\,\zeta_{i+1,j}^{n+1/2} = F(\psi^n)\,\zeta_{i,j-1}^{n} + G(\zeta)\,\zeta_{i,j}^{n} + H(\psi^n)\,\zeta_{i,j+1}^{n}$$

STEP 2

$$-F(\psi^n)\,\zeta_{i,j-1}^{n+1} + K(\zeta)\,\zeta_{i,j}^{n+1} - H(\psi^n)\,\zeta_{i,j+1}^{n+1} = -f(\psi^n)\,\zeta_{i-1,j}^{n+1/2} + L(\zeta)\,\zeta_{i,j}^{n+1/2} - h(\psi^n)\,\zeta_{i+1,j}^{n+1/2}$$

RESULTS IN TRIDIAGONAL MATRIX WHICH IS SOLVED BY THE THOMAS METHOD

$$\begin{pmatrix} \times\times & & & & \\ \times\times\times & & & & \\ \times\times\times & & & & \\ \times\times\times & & & & \\ & \ddots & & & \\ & & \times\times\times & & \\ & & \times\times\times & & \\ & & \times\times & & \end{pmatrix} \begin{pmatrix} \zeta \end{pmatrix} = \begin{pmatrix} BC \\ \\ \\ \\ BC \end{pmatrix}$$

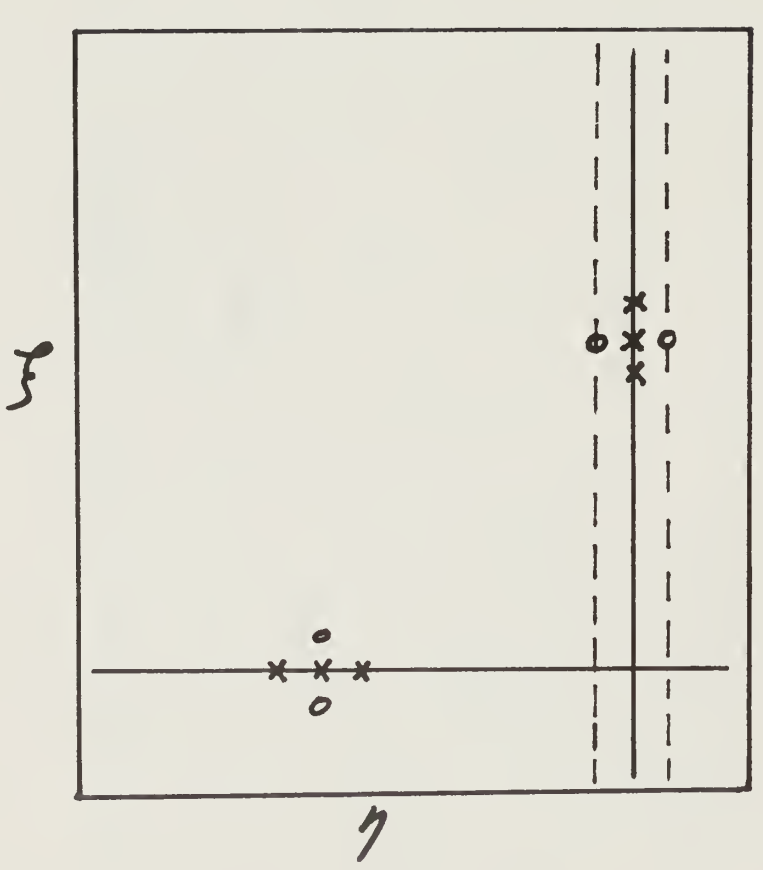
FIGURE 7

activity is a constant goal. Frequently the manner in which data is stored in PEM will determine how efficiently the computations can be carried out. For instance, solution of a tridiagonal matrix by the Thomas Method is inherently a serial process. When the solution is implicit in the columns the coefficient vectors for each column can be placed in a separate PE. Thus if the field width is choosen to be 62 (allowing two PE's for boundary conditions) then maximum use of the array processor can be obtained. The sequential processes take place as though 62 serial computers were calculating the result and thus 62 results are computed as fast as one.

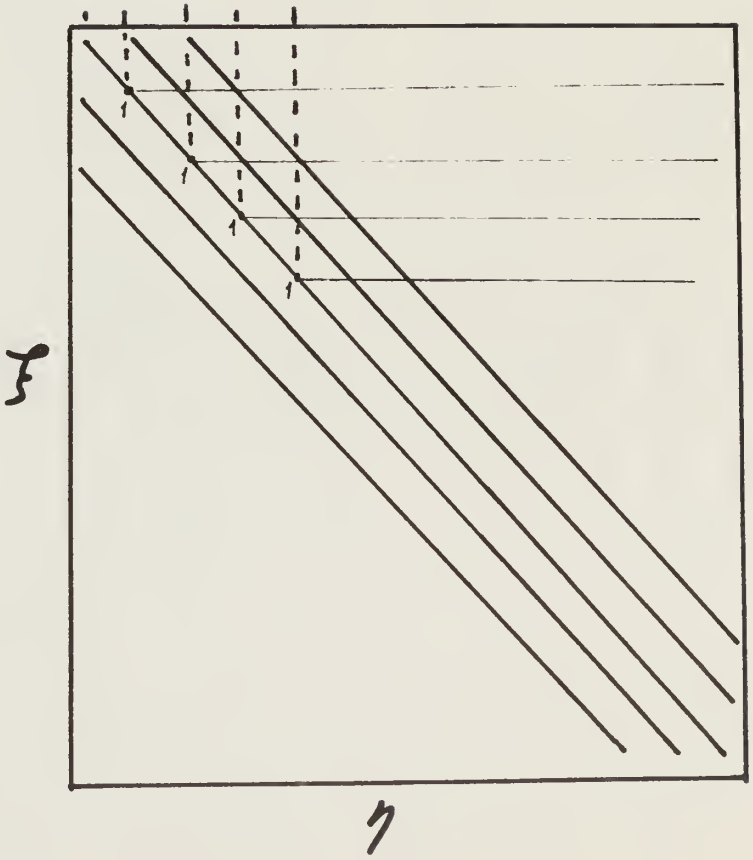
When the solution is implicit in the rows a problem arises. Since the process must follow a given sequence beginning with the first element in the row and continuing to the end of the row and then return, how can the first element of more than one row be accessed simultaneously? This problem is simply solved by shifting the coefficients in the second row one PE to the right, in the third row 2 PE's and so on until the entire field is skewed as shown in Figure 8. Then the first element in each of the 62 rows can be brought into separate PE's simultaneously. By appropriate indexing and routing of data among PE's, the solution for 62 rows is obtained simultaneously, thus using the array processor efficiently.

### 4.2.4 The stream function equation

Two different techniques were used to update the values of the stream function, an iterative method and a direct method. Son's algorithm used only an iterative method. Timing studies showed that updating the stream function was the most time consuming step in the algorithm and thus



DATA STORED STRAIGHT



DATA SKEWED

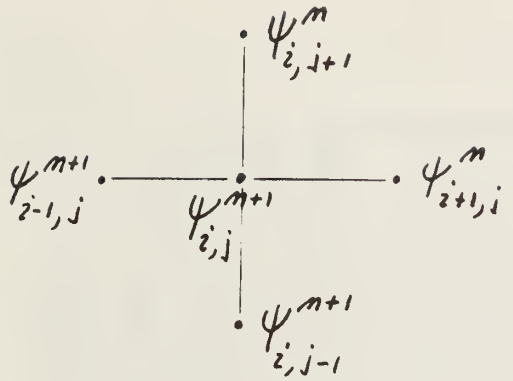
FIGURE 8

prompted a search for a faster method. The direct method appears to cut the computation time 50 to 60 per cent.

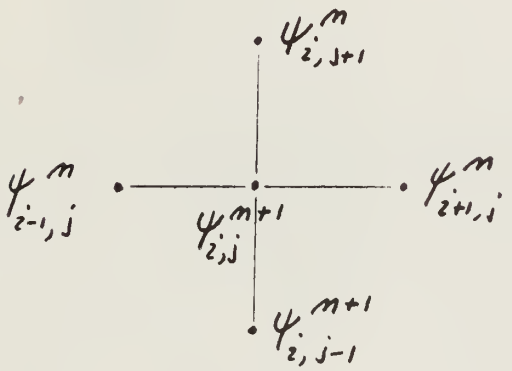
The iterative method used in Son's serial algorithm was the successive overrelaxation method (SOR). The computational molecule is shown in Figure 9-A and uses two updated values and two old values to determine the new center value. The serial algorithm thus starts in the lower left corner of the mesh and sweeps from left to right on each row, moving up the mesh.

In order to use ILLIAC efficiently, a whole row must be processed at once so that the 62 PE's are all active. The first scheme tried used the computational molecule in Figure 9-B. Only the point from the previous row was an updated value. Unfortunately this method did not converge.

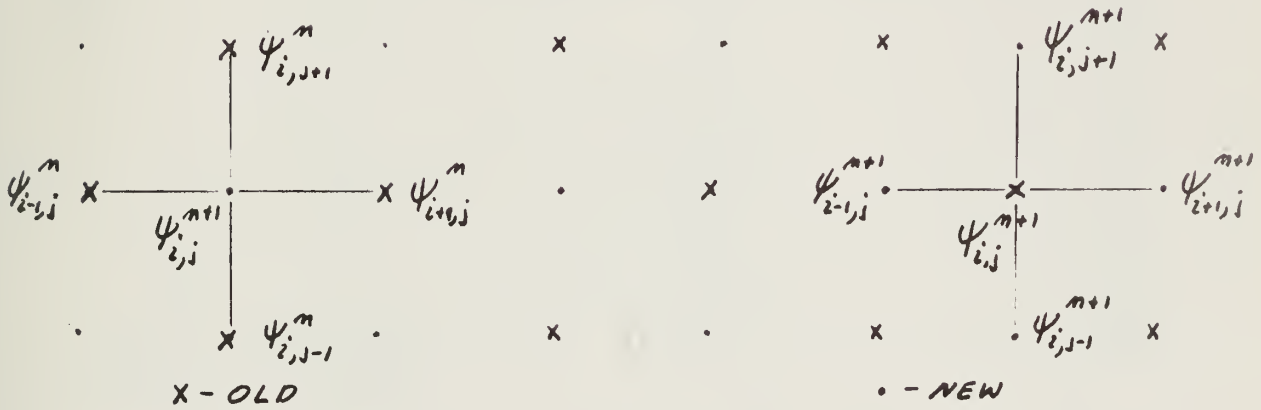
A slightly more complicated but rapidly convergent method was developed and will be called the modified successive overrelaxation method (MSOR). This method identifies each point in the mesh as either an odd point or an even point as determined by the sum of its indicies. The method requires two passes through the mesh. On the first, each odd point is updated using only old values as shown in the left computational molecule in Figure 9-C. The second pass then updates the even points using all new information as shown in the molecule on the right. This staggering of points along with the proper index modification allows all the odd or even points in two rows to be updated simultaneously. The finite difference equation and a sketch of the computation mesh are given in figure 10-A. The MSOR method has been shown to have the same asympotic rate of convergence as the SOR method. A Center document by J. Ericksen [2] treats this in more detail. The MSOR technique makes efficient use of the ILLIAC IV array processor.




A. POINT SOR



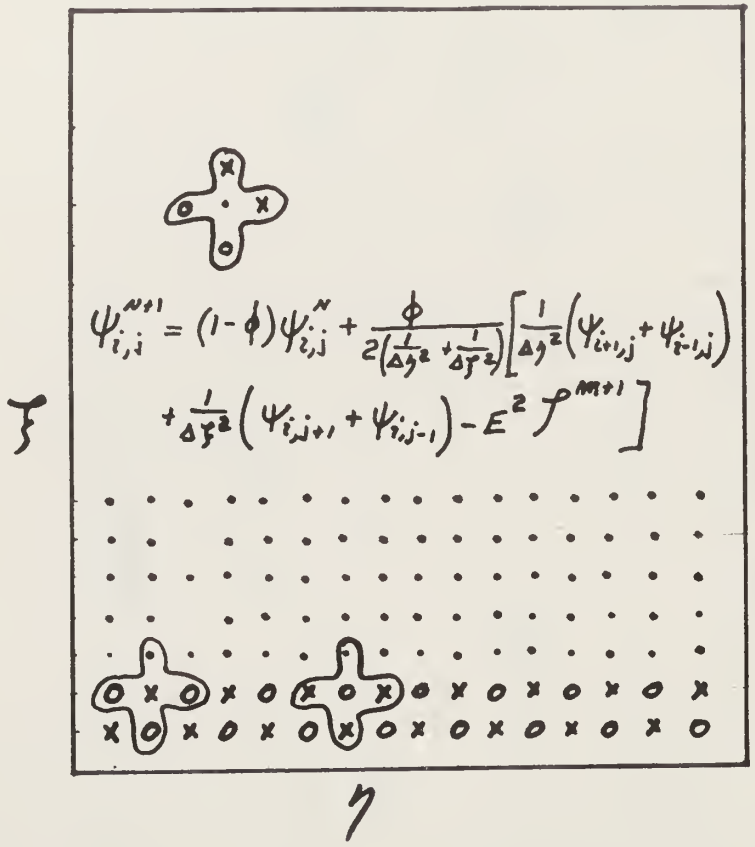

B. ROW SOR



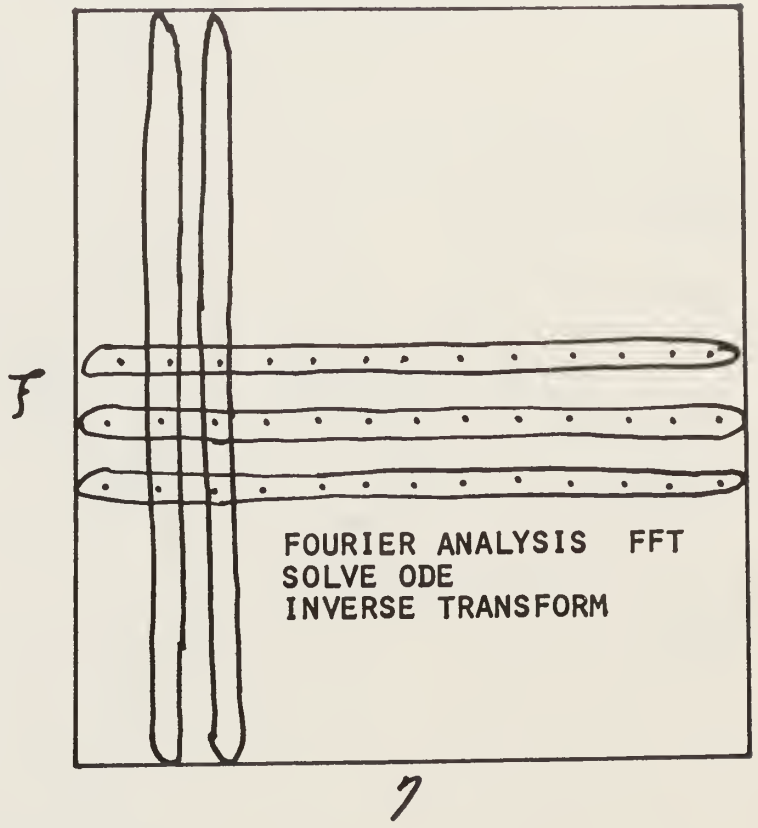

C. MSOR

FIGURE 9

SOLUTION OF THE POISSON EQUATION




A. ITERATIVE
MSOR




B. DIRECT
HOCKNEY

FIGURE 10

The stream function equation is Poisson's Equation and its iterative solution has received much attention. Recently Hockney [3] has proposed a technique for obtaining a solution by a noniterative method with a significant savings in computation time. Since the iterative solution of Poisson's Equation represented more than 80% of the total computation time for the problem, it was decided to try Hockney's method. This direct method limits the number of mesh points in each direction to $(2^n+1)$, where n is any integer. This is an undesirable feature but in most cases it is quite tolerable given the increased computational speed. The program was initially developed in ALGOL and then converted to GLYPNIR. The GLYPNIR program was developed by J. Ericksen and is discussed in more detail in his report [2]. The method Fourier transforms the vorticity based variables along a whole row, resulting in a series of Fourier coefficients which are independent from column to column (PE to PE) but are implicit up the columns. The computational mesh is shown in Figure 10-B. In effect the Fourier analysis has reduced the problem from one of solving a partial differential equation to one of solving a series of coupled ordinary differential equations. This later solution is accomplished by inverting a simple tridiagonal matrix. The resulting inverse Fourier components are then converted back by Fourier synthesis to yield the stream function distribution. The method reportedly uses ILLIAC efficiently and thus should represent a savings of time over the iterative method.

Test programs for each method of updating the stream function have been prepared in GLYPNIR. The iterative program is titled CICHY/GLYSOND and the direct program is called CICHY/GLYDIRECT.

## 4.3 Programming the Algorithm

The strategy used in programming this algorithm was to use subroutines to represent separate logical steps in the algorithm. The subroutine MAIN controls the program, calling other routines as needed. The subroutine names are descriptive of the function they perform.

The program is designed to handle up to 62 internal mesh points across the field and up to 254 internal mesh points along the field. Boundary conditions are inserted outside these points to complete the definition of the field.

The program parameters are specified in the subroutine SYSTEM PARAMETERS. They are the Reynolds Number (NREY), the dimensionless time increment (DT), the dimensionless time limit for the problem (TLMT), the error limit for the convergence of the iterative solution of the stream function equation (LIMIT), the total (includes boundary) number of mesh points in the circumferential direction (NETA), the total number of mesh points in the radial direction (NTSI), the maximum value of the circumferential coordinate (ETAMAX), the maximum value of the radial coordinate (TSIMAX), and the increment in time steps at which printouts are to be taken (PINC). Given these parameters,the program generates all the other information and control parameters it needs to solve the problem.

The two GLYPNIR programs developed are listed in Appendicies A and B. These programs contain many comments indicating how the actual operations of the algorithm are carried out in the computer.

## 4.4 Testing the Algorithm

GLYSOND is coded in GLYPNIR and has been successfully compiled on the GLYPNIR Compiler. The ASK produced from the compilation has been

successfully compiled into machine language. This machine language has not been tested on the ILLIAC IV. However, a simulator for the ILLIAC IV hardware has been developed and runs on a Burroughs B6700. With this simulator, machine code can be tested just as if it were running on ILLIAC IV. The major problem with testing code on the simulator is that, since the simulator runs in a serial manner, it runs much slower than ILLIAC , in fact the ratio is 200,000 to one.

It was not possible to test the machine code from GLYSOND for a reasonably sized mesh. Therefore, a 9x17 mesh was chosen and the number of iterations in the iterative MSOR was limited to six. Even so, to accomplish three time steps on the simulator required 12 minutes of CPU time.

In order to check the results of this abbreviated mesh problem it was necessary to develop a serial solution to the problem. Son's FORTRAN program was converted to an ALGOL program titled CICHY/MASTER, Appendix C. MASTER was used on a 31x51 mesh at Reynolds Number of 40 and the results checked well with those of Son. The same program was then used to generate results for 3 time steps using a 9x17 mesh and with iterations in the SOR limited to 6. These results compared well with the results from GLYSOND. This close comparison was taken as proof that the GLYPNIR program worked well and was ready to be tested on ILLIAC IV.

The same test procedure was applied to the program using the direct solution of the stream function equation. The GLYPNIR program is titled CICHY/GLYDIRECT, Appendix B, and the ALGOL program is CICHY/DIRECT, Appendix D.

In order to have numerical results to check the solution obtained from GLYDIRECT on ILLIAC IV, DIRECT was used to solve the problem on

a 33x65 node mesh. These numerical results should compare closely with the numerical results obtained from GLYDIRECT run on ILLIAC IV using a similar sized mesh if the program is operating correctly. These numerical results are currently stored with the numerical results of Son at the Department of Chemical Engineering.

The next step in testing GLYSOND and GLYDIRECT is to run them on ILLIAC IV. This should be possible at least by June 1973. GLYSOND as shown in Appendix A is set up to solve a 9x17 mesh and limit MSOR to six iterations. A copy of this program is now resident in the TENEX file system at Ames and John Gaffney and Marvin Graham will attempt to run this as soon as the ILLIAC is ready. These results should compare well with those obtained from the simulator (the printouts are stored at the Department of Chemical Engineering). The next step should be to remove cards 153 and 597 from GLYSOND which will allow MSOR to iterate to convergence. Then run GLYSOND with NREY=40, NETA=31 and NTSI=51, and compare these results with those obtained by Son. CPU time should be requested so that comparisons of running time and accuracy can be made.

After GLYSOND has been tested, GLYDIRECT should receive similar testing. Because of changes in the GLYPNIR compiler one or two variable names in GLYDIRECT may now be reserved words. The compiler will point these out and they should be changed.

At present the final I/O specification are still not complete. It is expected that they should be finished soon and that as soon as they are finished a form of GLYPNIR I/O for ILLIAC IV will be implemented.

## 4.5 Problems with this Symmetric Formulation

There are several features of the symmetric formulation of the problem which indicate that a more general approach is desirable.

The case of symmetric vorticies is realized in practice only up to a Reynolds number of about 40. Above this Reynolds number the flow breaks down into a pattern of vortex shedding. The symmetric formulation cannot represent this flow pattern and is thus limited in its range of application.

The use of circular cylindrical coordinates limits the extent to which the downstream boundary can be moved away from the cylinder. The parallel flow assumption made at this boundary is fairly restrictive and it is important that it have a negligible effect on the size and shape of the trailing vorticies and on the pressure distribution around the cylinder. Due to the diverging radial coordinate lines, an unreasonable number of mesh points must be added to retain adequate downstream resolution as the boundary is moved away from the cylinder.

The mesh spacing in this formulation follows a strict geometric pattern, modified slightly to better account for the boundary layer. In fact, there are regions other than the boundary layer where variable gradients may be large but which are not adequately covered with mesh points, such as the shear layer at the near edges of the vorticies. It would be desirable to be able to tailor the mesh size in any region to the size of the gradients expected in that region. This control over mesh size would require a significantly more complicated program.

## 4.6 Conclusions

The ability of a parallel processor to solve one particular type of numerical fluid dynamics problem has been established with the completion and testing of the GLYSOND and GLYDIRECT parallel programs. With the proper choice of the number of mesh points, these algorithms should operate efficiently and result in very rapid solutions. Actual testing on ILLIAC IV will be required to determine just how much faster the parallel program is than the serial program.

The placement of data in the PEM was shown to be very important in maintaining a high efficiency during certain portions of the program. The routing overhead required is generally negligible, except possibly in the case of very short rows.

The programming of a parallel machine is more difficult than a serial machine, but given a good understanding of the machine architecture and a working knowledge of both programming languages, the programming of an algorithm such as this one is not unreasonably difficult. Planning of data storage areas, enabling mode control, and data movement patterns represent some of the more difficult tasks.

Some of the most difficult problems encountered in the development of the symmetric algorithm related to the uncertainties in the supporting software and the unreliability of local hardware. Hopefully both of these situations have been improved to the point where they are no longer serious problems.

## 5. THE ASYMMETRIC FLOW AROUND A CYLINDER

### 5.1 Description of the Problem

The asymmetric flow around a cylinder is of particular interest since flows of this type are frequently observed in actual fluid flow situations. The famous von Karmen vortex street is an example of asymmetric vortex shedding behind a cylinder.

One objective of this second study, besides relaxing the conditions of symmetry, was to find a coordinate system which would allow the downstream boundary condition to be moved away from the cylinder while not increasing unreasonably the number of mesh points needed to maintain resolution. This constraint suggested that some kind of a rectangular coordinate system

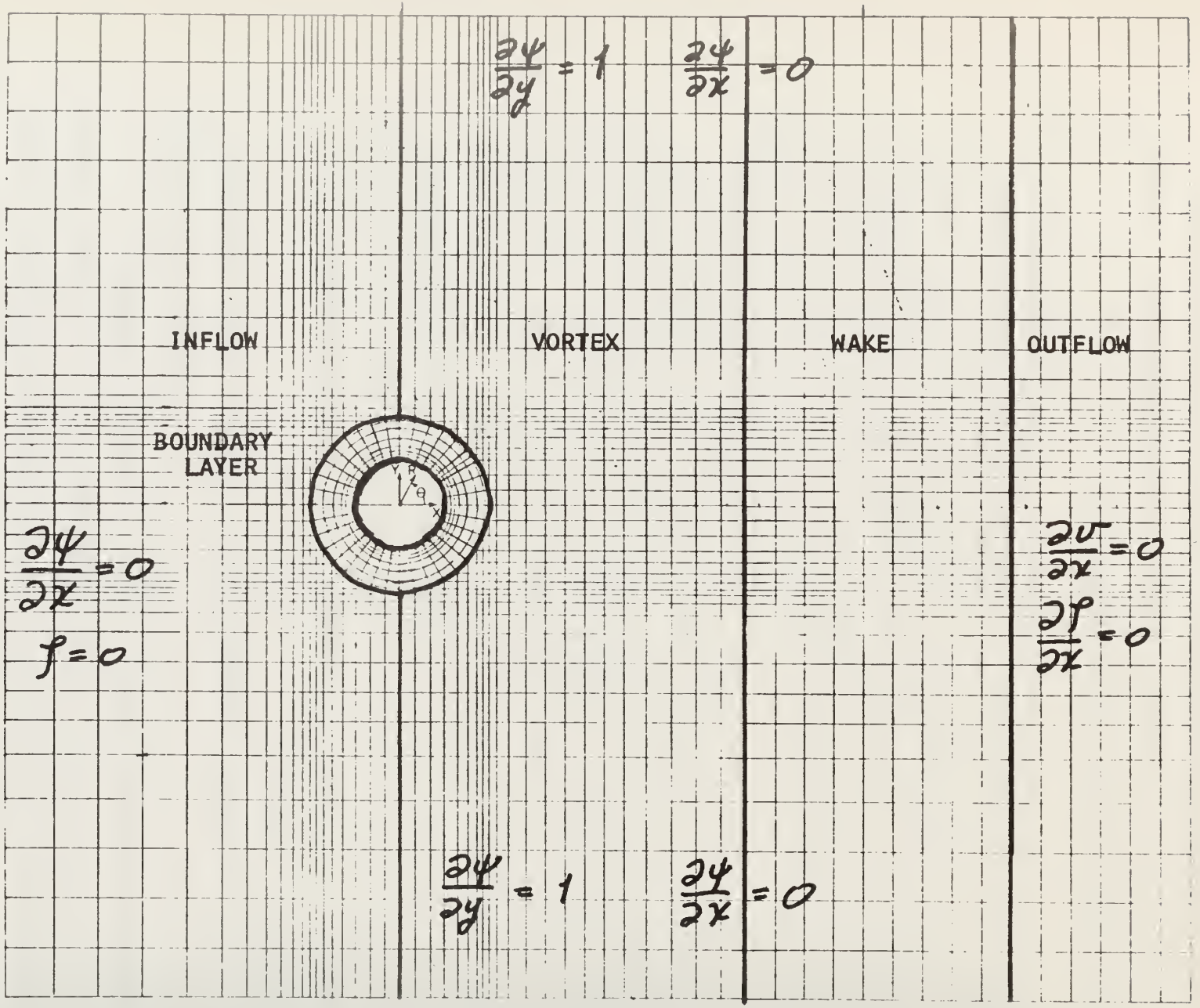
would be appropriate downstream. However, near the surface of the cylinder, where resolution is needed for the boundary layer, a circular cylindrical coordinate system seemed best.

The search for a coordinate system which would be cylinder like near the cylinder surface and rectangular downstream led to the trial of a set of coordinates based on the potential flow solution. These conformed well to the shape of the cylinder and became rectangular at large distances from the cylinder surface. These seemingly perfect coordinates had, however, one fatal flaw, a singularity at the leading and trailing stagnation points. After considering several methods to circumvent the singularity these coordinates were abandoned in favor of the combination of circular and rectangular coordinates proposed by D. C. Thoman and A. A. Szewczyk [5].

The theoretical development used for the asymmetric problem has been taken directly from the report mentioned above. The equations to be solved are basically the same as those developed in Son's work but the boundary conditions are slightly different since the flow is asymmetric. The boundary conditions and the organization of the flow field are shown in Figure 11.

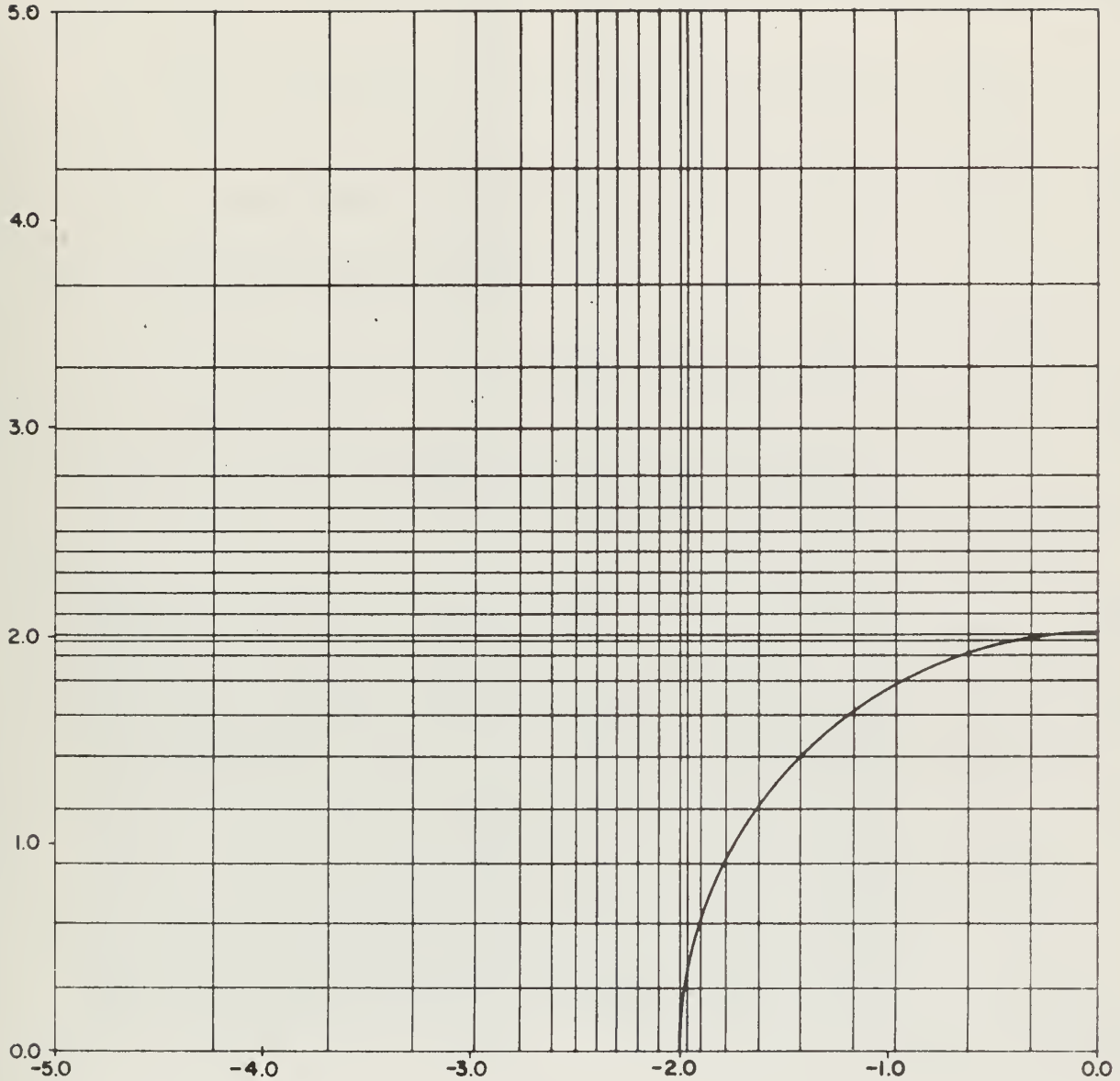
The five regions shown in Figure 11 represent areas of differing fluid activity. The mesh distribution in each region has been carefully chosen to adequately resolve the expected variable gradients.

The mesh distribution over the important parts of Regions 2 and 3 is illustrated in Figures 12 and 13. Although these drawings are not exact, they show the increased concentration of mesh points just behind the cylinder, especially in the region where a shear layer might be expected. Fewer points are placed in Region 2, upstream of the cylinder, since the flow is expected to be quite regular there.



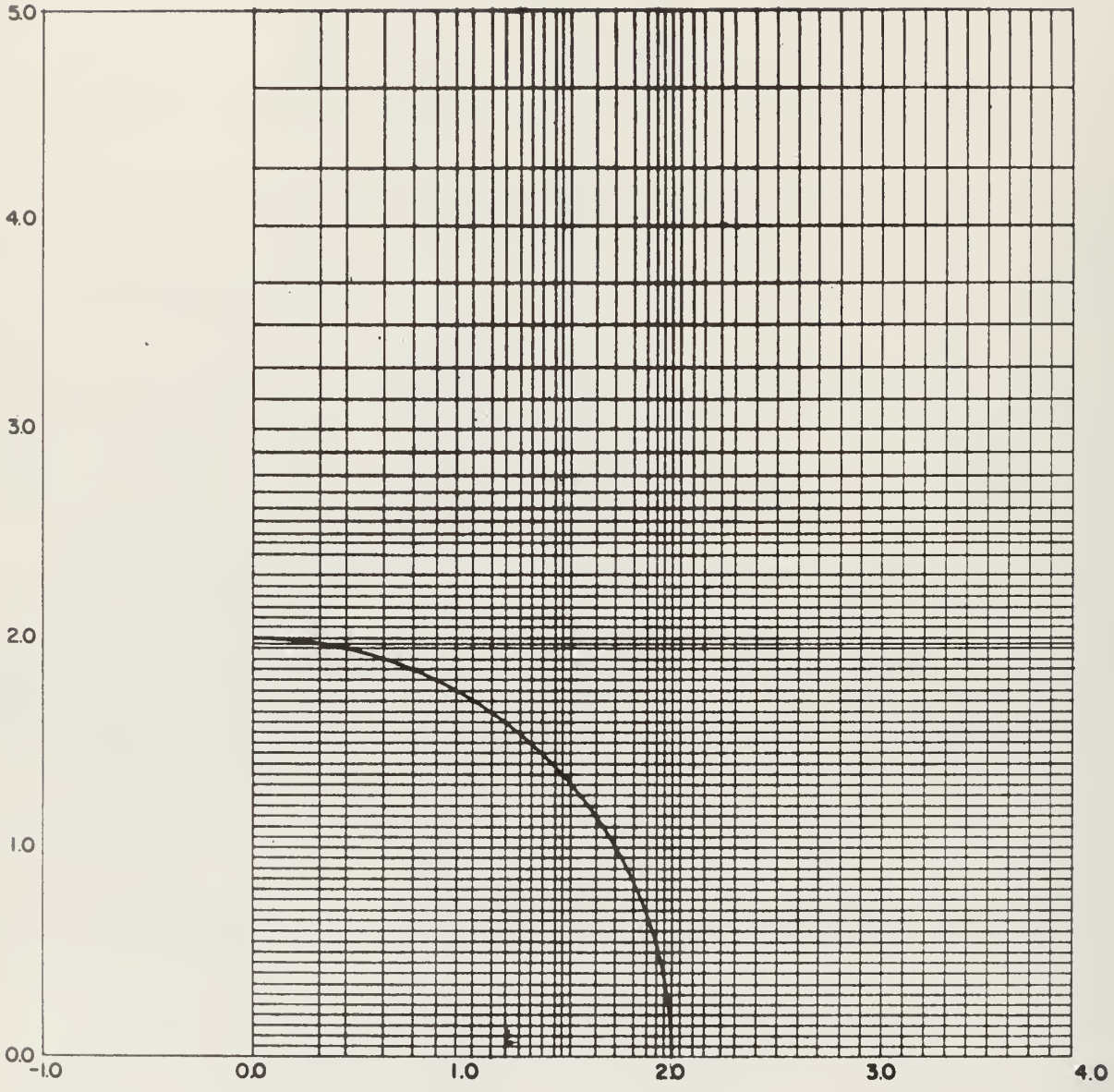

FIVE FLOW REGIONS AND BOUNDARY CONDITIONS

FEIGURE 11




PART OF REGION 2

FIGURE 12




PART OF REGION 3

FIGURE 13

The relative sizes of the lateral mesh spacing used in Regions 3, 4 and 5 are shown in Figure 14. The length of Regions 4 and 5 can be changed easily to assess the effect of the downstream boundary condition on the solution.

The mesh spacings used in the cylindrical coordinates of Region 1 are illustrated in Figure 15. The points at R=2 correspond to points in the rectangular meshes of Regions 2 and 3.

## 5.2 Development of the Algorithm

The general algorithm for the solution of the asymmetric problem follows the same pattern as discussed for the symmetric problem. A list of the steps in the algorithm are given in Figure 16.

Two sets of finite difference equations were required, one in rectangular coordinates and the other in cylindrical coordinates. The finite difference forms of the vorticity transport equation and the stream function equation for rectangular coordinates are shown in Figure 17. The equations are explicit and allow for a variable mesh size through the inclusions of the cell width $b_i$ and cell height $a_j$.

The presence of the circular coordinate system in the interior of the rectangular system excludes the use of implicit techniques. It is, therefore, necessary to devise a finite difference form of the vorticity transport equation which is explicit. This form requires that two components of the fluid velocity, u and v, be computed from the stream function distribution before the vorticity values at the next time level can be obtained. The implicit methods used to solve for the vorticity in the symmetric problem were stable for almost any reasonable time step. The stable time step for the explicit method is generally smaller, thus increasing the computation time required to obtain a given solution.

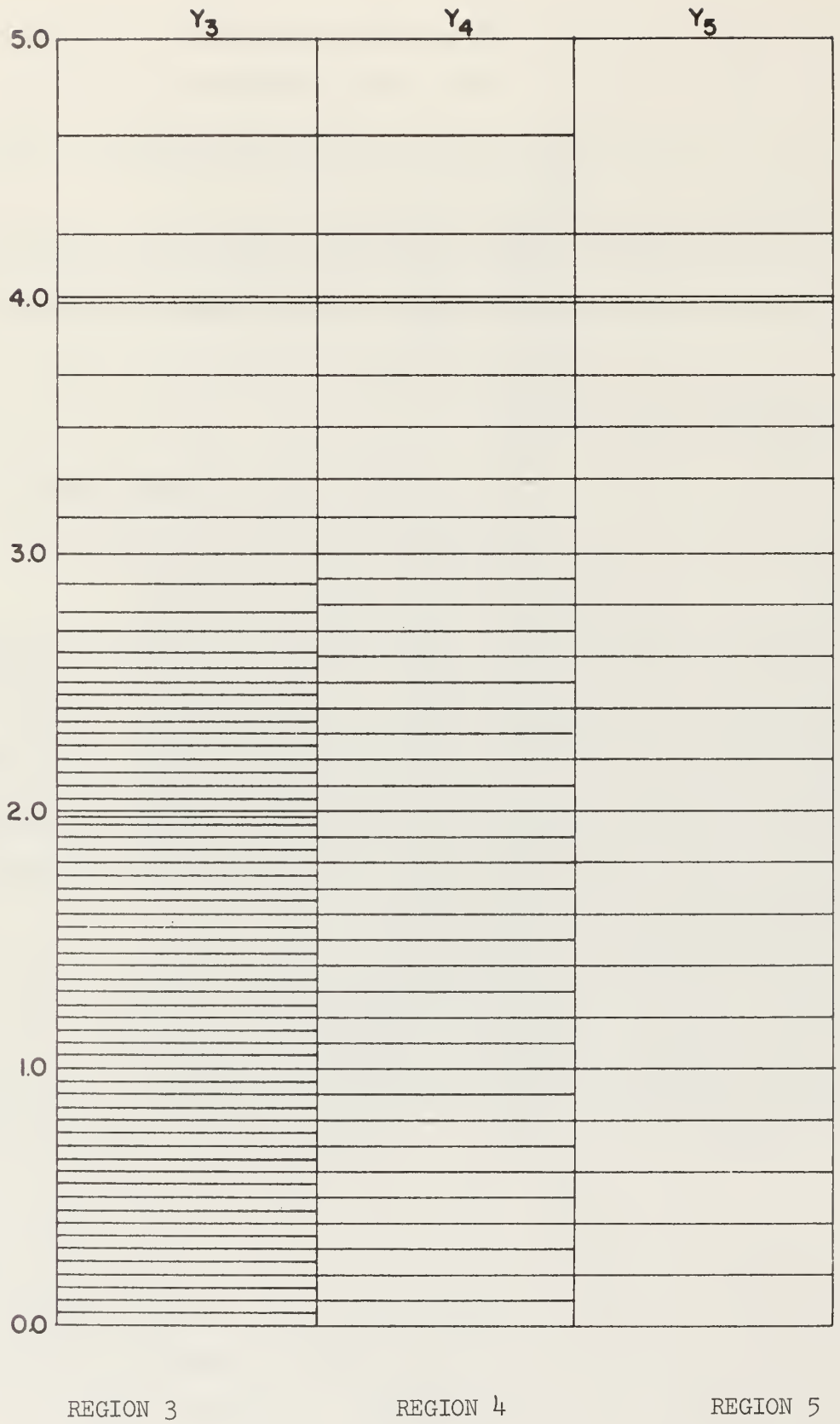
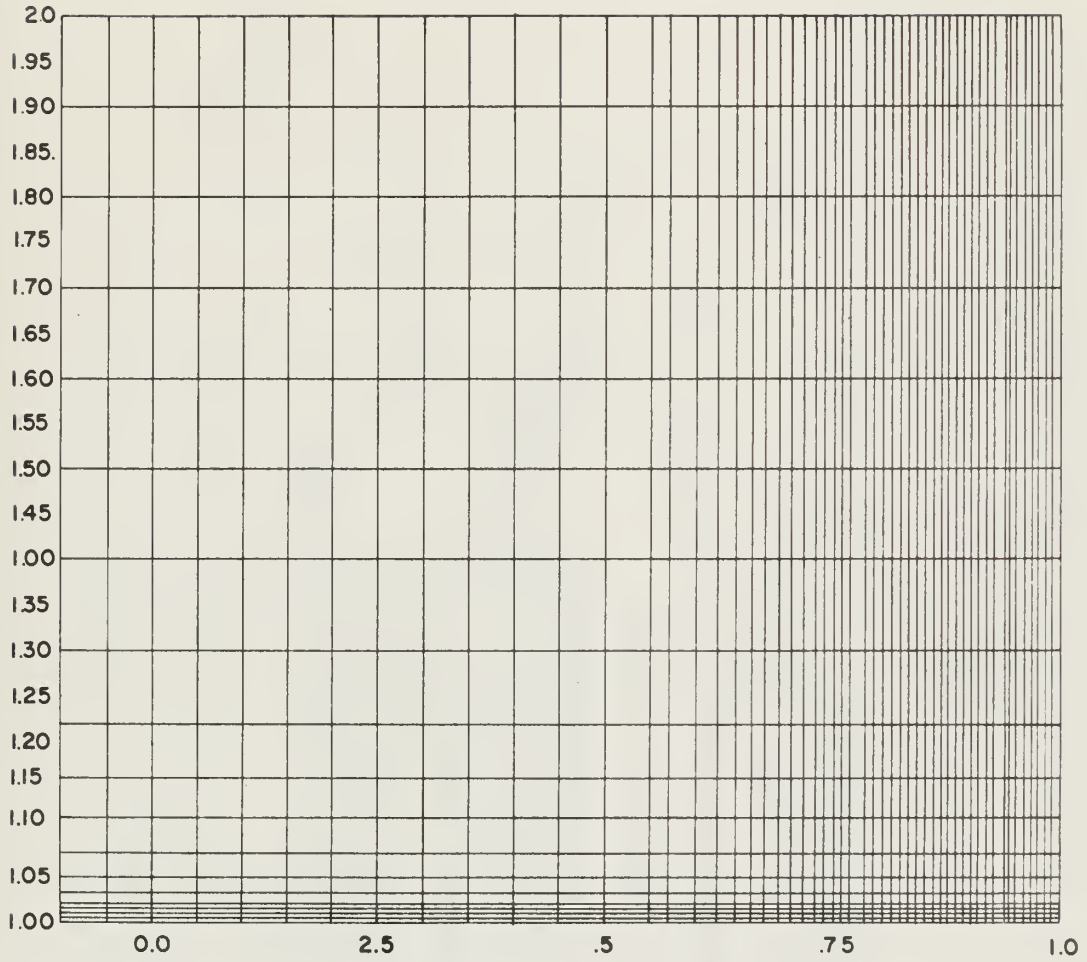



FIGURE 14




REGION 1

FIGURE 15

COMPUTATIONAL ALGORITHM

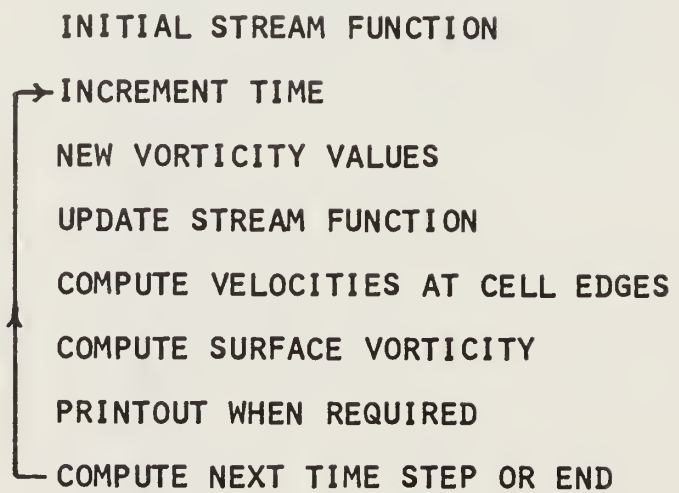



FIGURE 16

VORTICITY TRANSPORT EQUATION IN DIFFERENCE FORM

$$\frac{\zeta_{i,j}^{K+1} - \zeta_{i,j}^{K}}{\Delta\tau} + \frac{(u\zeta)_{i+1/2,j}^{K} - (u\zeta)_{i-1/2,j}^{K}}{b_i} + \frac{(v\zeta)_{i,j+1/2}^{K} - (v\zeta)_{i,j-1/2}^{K}}{a_j}$$

$$= \partial \frac{1}{b_i}\left[\frac{\zeta_{i+1,j}^{K} - \zeta_{i,j}^{K}}{0.5(b_{i+1} + b_i)} - \frac{\zeta_{i,j}^{K} - \zeta_{i-1,j}^{K}}{0.5(b_i + b_{i-1})}\right]$$

$$+ \partial \frac{1}{a_j}\left[\frac{\zeta_{i,j+1}^{K} - \zeta_{i,j}^{K}}{0.5(a_{j+1} + a_j)} - \frac{\zeta_{i,j}^{K} - \zeta_{i,j-1}^{K}}{0.5(a_j + a_{j-1})}\right]$$

STREAM FUNCTION EQUATION IN DIFFERENCE FORM

$$\psi_{i,j} = \frac{2}{d_{i,j}}\left[\frac{\zeta_{i,j}}{2} + \frac{\psi_{i-1,j}}{b_i(b_{i-1} + b_i)} + \frac{\psi_{i+1,j}}{b_i(b_i + b_{i+1})}\right.$$

$$\left. + \frac{\psi_{i,j-1}}{a_j(a_{j-1} + a_j)} + \frac{\psi_{i,j+1}}{a_j(a_j + a_{j+1})}\right]$$

FIGURE 17

Thoman and Szewczyk discuss the stability requirements for the explicit method in their report.

The difference approximation to a partial differential operator on a nonuniform mesh has a larger truncation error than that on a uniform mesh of similar cell size. The additional error arises due to the fact that the coefficient of the first neglected term in the Taylors series approximation to the derivative contains the difference of the mesh sizes on each side of the point of interest. When the mesh size is uniform, this difference is zero and the effective accuracy of the approximation is increased by one order. However, in the case of a nonuniform mesh, this term must be retained and it contributes to the error in the approximation.

The importance of the error conditions mentioned above can be minimized when taken over the whole mesh by keeping the cell sizes constant wherever possible. When moving from a region of one cell size to that of another, the amount of cell to cell variation should be limited to an amount such that the coefficient of the first neglected term is always nearly zero. In addition, if the regions of changing cell size can be placed in areas where the value of the first neglected derivative itself is expected to be small, the error encountered will be even smaller.

Another technique to reduce error has been proposed but was not used in this study. After applying the explicit form to compute the values at the next time step, use the finite difference forms of the neglected derivatives to evaluate the error caused by truncating the Taylors series approximation. Subtract this error from the computed values and repeat the process until the calculated truncation error is reduced to zero. This technique would improve the accuracy of the symmetric and asymmetric solutions and consideration should be given to applying it to these problems.

## 5.3 Programming the Algorithm

The program for the asymmetric flow problem is coded in the latest version of GLYPNIR. Appendix E contains the sections of code which have been completed at this time. Further work on the program will be needed before it can be run on ILLIAC IV. The following sections of the report discuss some of the problems encountered in programming this large code for ILLIAC, and describes some possible solutions. No attempt to optimize the code has been made at this point in its development.

### 5.3.1 The buffer system

The symmetric problem was small enough to be core contained. The asymmetric case, however, was too large to fit in the PEM and thus a buffer system was developed.

The total number of rows of PEM storage required to hold the entire stream function mesh was first determined. The total was then divided into four sections each containing nearly the same number of rows. This division was made strictly on the basis of the number of rows, without regard to the boundaries of the various flow regions. The section sizes were then doubled to account for the vorticity storage needed. A separate disk file was set up to hold each of the 4 sections. Two areas were reserved in the PEM, each area large enough to hold one section. The Buffer System is illustrated in Figure 18.

During the course of the computation, a section is read into or written out of one area while calculations proceed in the other. When calculations and transfers are complete, calculation moves to the second area and transfers begin on the area just calculated. Because of ILLIAC 's great speed, it is expected that the transfer times will exceed the calculation time in most problems, a condition known as being I/O bound.

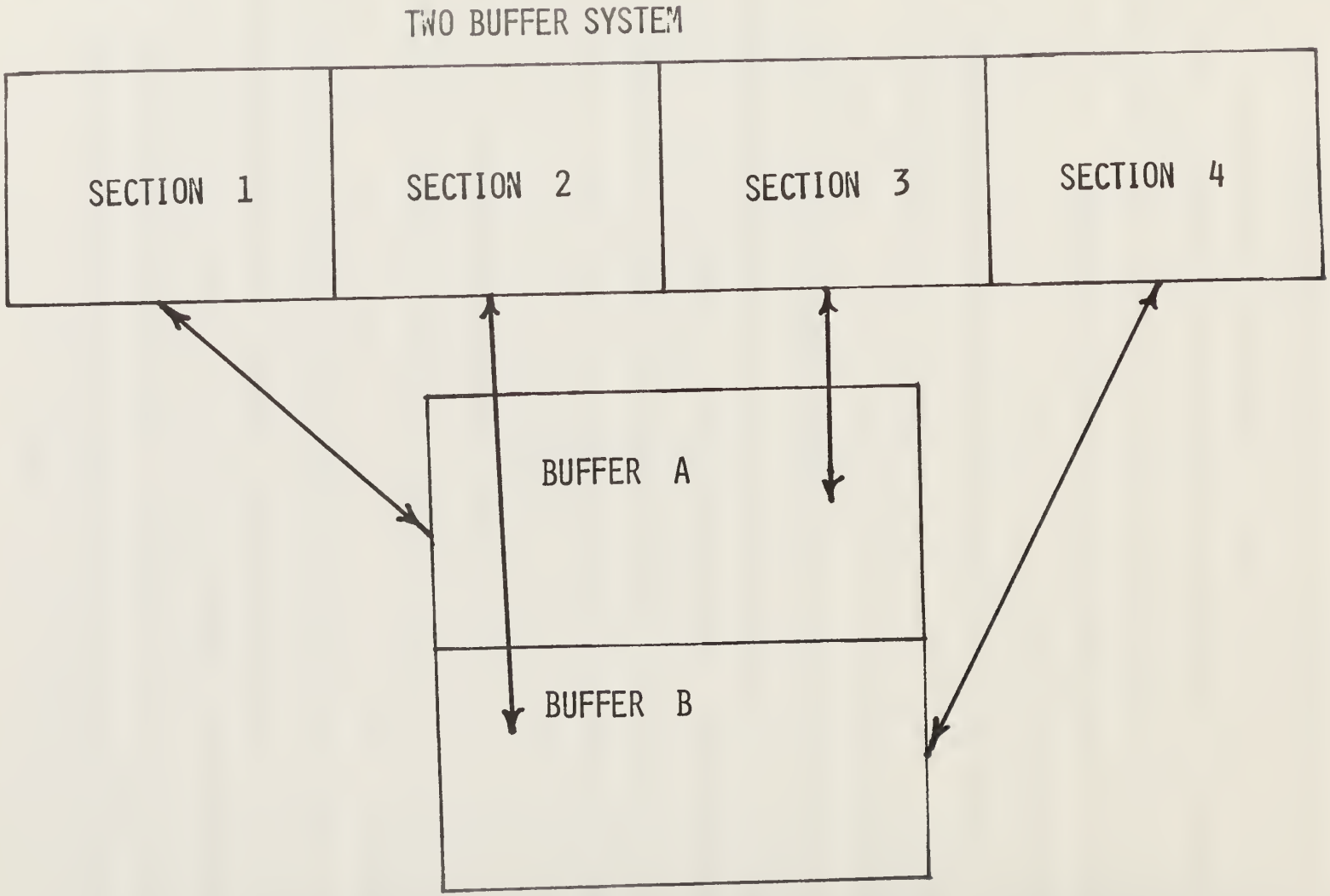

TWO BUFFER SYSTEM
SECTION 1
SECTION 2
SECTION 3
SECTION 4
BUFFER A
BUFFER B

FIGURE 18

A working row (or rows) was designated outside of the transferable areas in PEM to assist in the transportation of information from one section to the next. When calculations are complete in a section, the last row in that section is moved to the work area. The first operation on the next section is to insert the new information from the work area into the first row. The calculations can then proceed as usual.

The sectioning of the program and the use of two buffers suggested that a separate piece of code could be used to carry out the calculations in each section. This idea has been implemented through the use of a separate subroutine for each section. The technique provides a logical organization to the program. It is particularly useful in this problem as it allows a fairly complicated program to be broken down into a series less complicated units.

### 5.3.2 Organization of PEM

The arrangement of the mesh points in PEM is a very important operation in setting up the program. It is necessary to lay out each buffer as a separate unit, indexed all the way through with respect to its first row. Figures 19 through 22 present the allocation of PEM storage for each buffer load. The numbers along the right edge are the buffer indicies. On their right are the coordinate value of the mesh line stored in that PEM row. The numbers on the left represent the regional indicies, indexed relative to the first row in the region. All of these indicies are used in the program.

The number of data points in a mesh row exceeds 64 in several regions. This makes necessary the sectioning of the row into pieces,

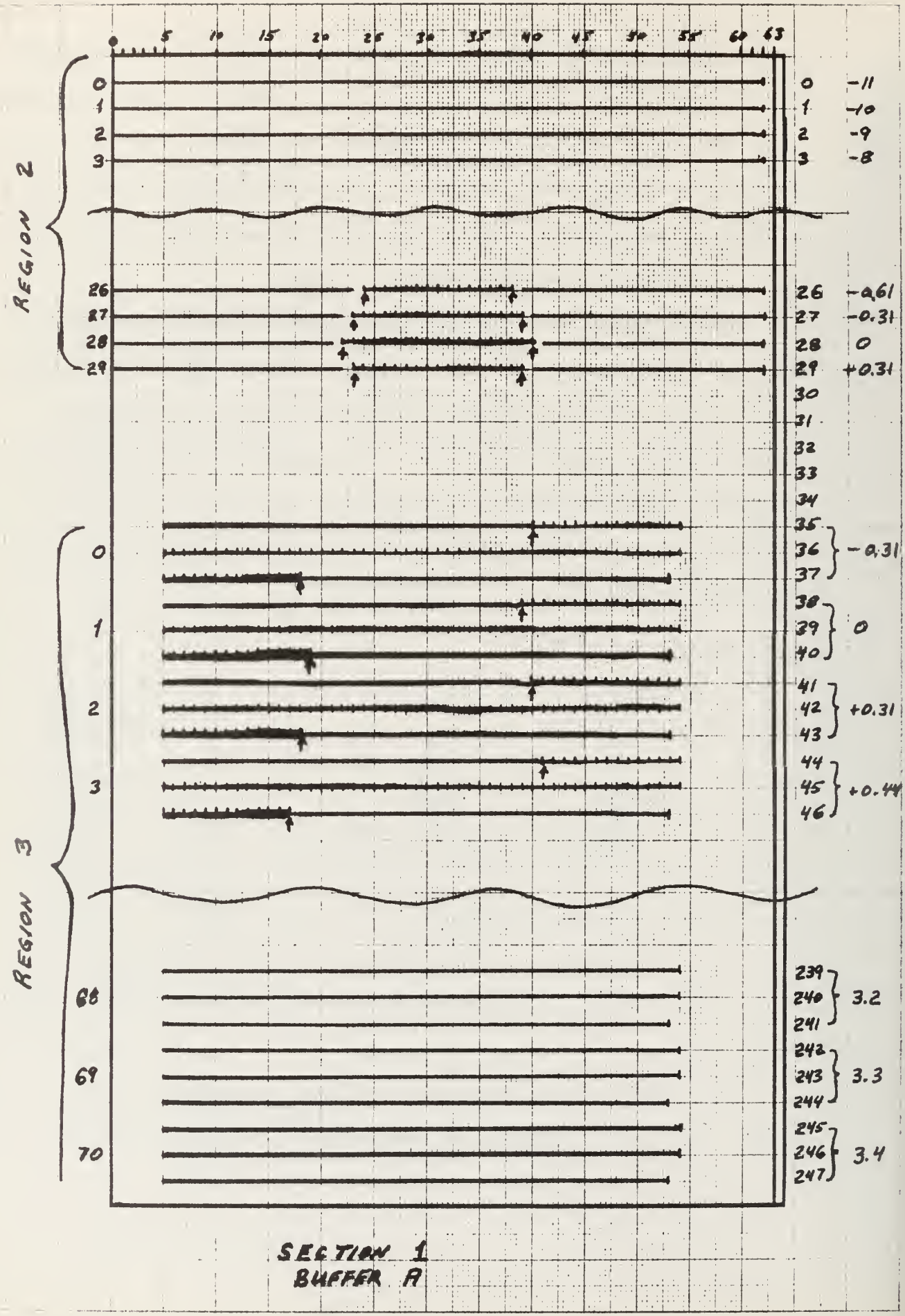

5 10 15 20 25 30 35 40 45 50 55 60 63
REGION 2
0
1
2
3
26
27
28
29
REGION 3
0
1
2
3
68
69
70
0 -11
1 -10
2 -9
3 -8
26 -0.61
27 -0.31
28 0
29 +0.31
30
31
32
33
34
35
36 -0.31
37
38
39 0
40
41
42 +0.31
43
44
45 +0.44
46
239
240 3.2
241
242
243 3.3
244
245
246 3.4
247
SECTION 1
BUFFER A

FIGURE 19

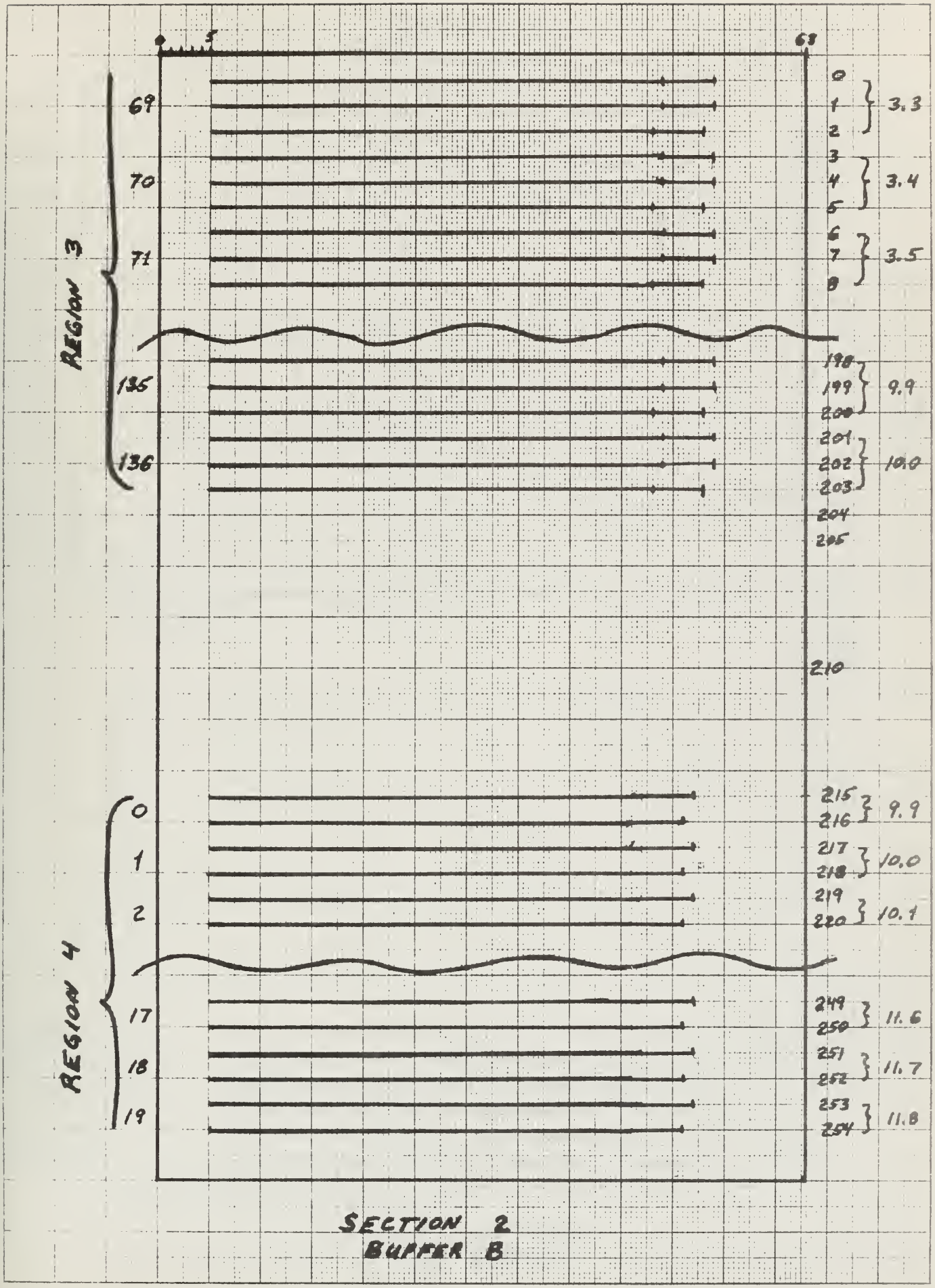

1
68
REGION 3
69
70
71
135
136
0
1
2
3
4
5
6
7
8
198
199
200
201
202
203
204
205
210
3.3
3.4
3.5
9.9
10.0
REGION 4
0
1
2
17
18
19
215
216
217
218
219
220
249
250
251
252
253
254
9.9
10.0
10.1
11.6
11.7
11.8
SECTION 2
BUFFER B

FIGURE 20

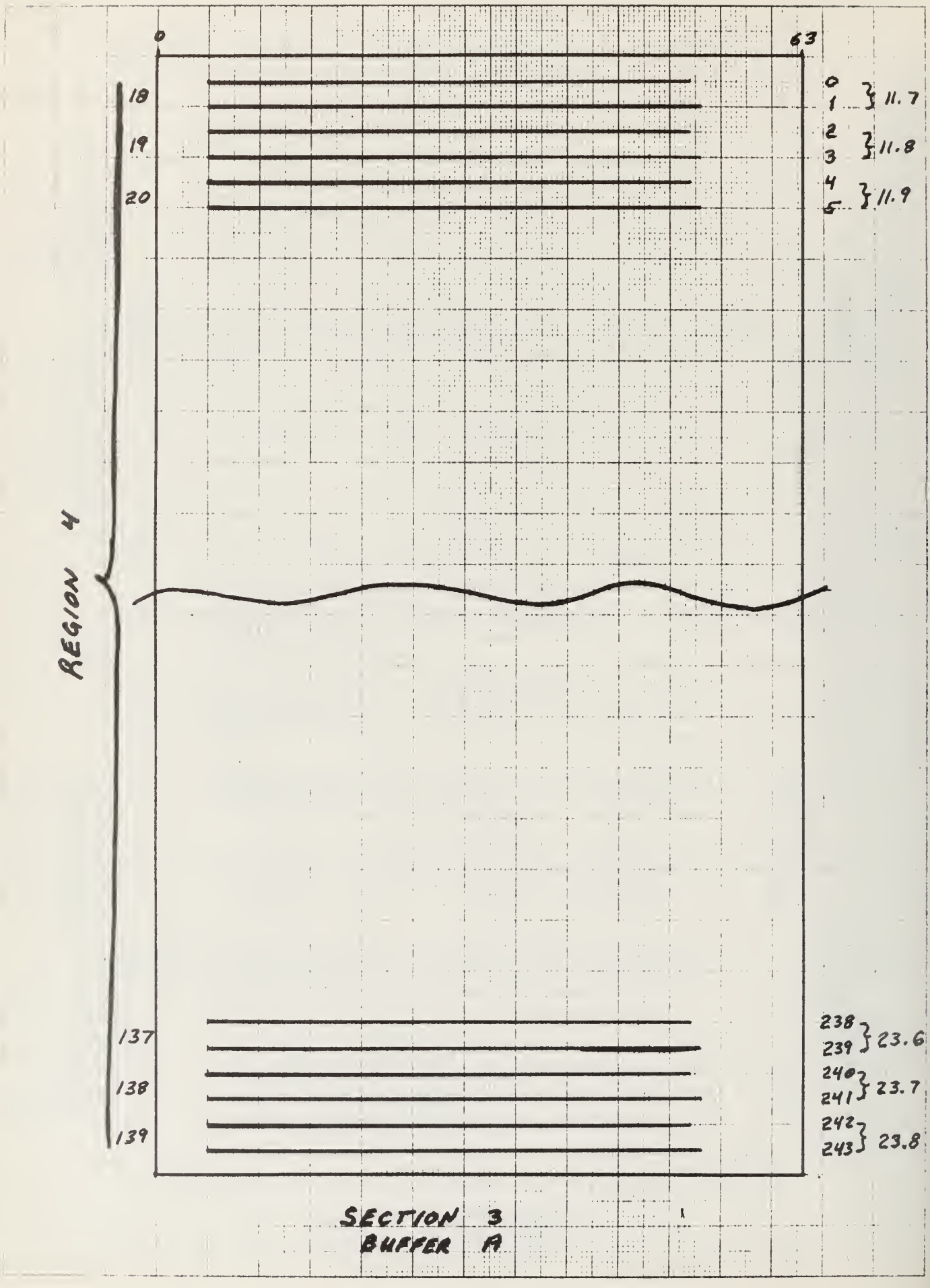

0
63
18
19
20
REGION 4
0
1
11.7
2
3
11.8
4
5
11.9
137
138
139
238
239
23.6
240
241
23.7
242
243
23.8
SECTION 3
BUFFER A

FIGURE 21

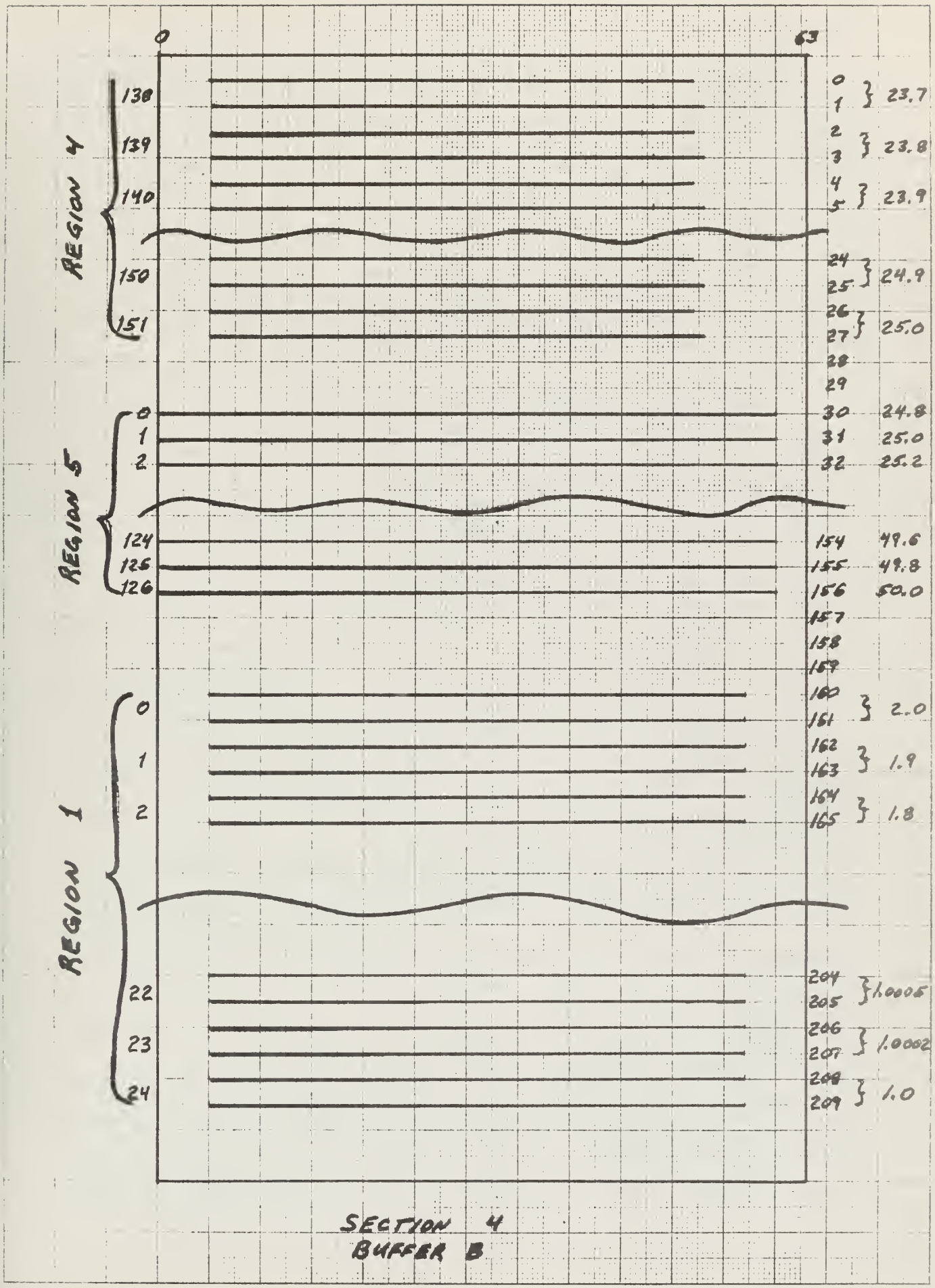

0
63
REGION 4
138
139
140
150
151
REGION 5
0
1
2
124
125
126
REGION 1
0
1
2
22
23
24
0
1
2
3
4
5
24
25
26
27
28
29
30
31
32
154
155
156
157
158
159
160
161
162
163
164
165
204
205
206
207
208
209
23.7
23.8
23.9
24.9
25.0
24.8
25.0
25.2
49.6
49.8
50.0
2.0
1.9
1.8
1.0005
1.0002
1.0
SECTION 4
BUFFER B

FIGURE 22

each of which will fit into a PE row. This sectioning results in complicated indexing problems. The sketches shown in Figures 19 to 22 are very useful in keeping the indexing straight. It is also necessary to leave at least one working position each side of the split in the row and to exchange data between working spaces as the computation proceeds from one PE row to another within a mesh row. This does cut into the efficiency of the calculations but it is difficult to assess how much.

The boolean mode variables control the enabling and disabling of PE's in the array. Each row has its own mode pattern. These patterns are carefully constructed so as to define exactly the active mesh. Separate mode patterns must be used when boundary conditions are evaluated. The data allocation sketches are very helpful in setting up the proper mode control patterns.

The nontransferable portion of PEM contains a number of variables which are associated with specific mesh points. The principle variables are the coordinate values of each mesh point, the cell width and height of each mesh point and the modes for each row. These are all assessed by regional indicies and are given a variable name which indicates to which region they belong. The coordinates and cell sizes must be provided by the user before the program is compiled. A separate procedure could be developed to obtain the cell sizes from the coordinates but that is not implemented in this program.

### 5.3.3 Boundary conditions

The boundary conditions which involve the endpoints of PEM rows require special attention. Each point must be treated separately, as in a serial machine. Since the endpoints are a small fraction of the total

mesh points, this inefficiency may not be too serious. The boundaries which involve whole PE rows are handled with little loss of efficiency.

The downstream boundary conditions used in this problem were taken directly from the report of Thoman and Szewczyk. They are that the axial velocity gradient and the axial vorticity gradient are both zero. These boundary conditions were selected from those experimentally tested as the ones which produced the most desirable solution. These conditions should be used in the initial version of this program but other more Physically realistic conditions should be examined in the future. A boundary layer type of outflow condition with $\frac{\partial^2 \zeta}{\partial x^2} = 0$ and $\frac{\partial^2 \psi}{\partial x^2} = 0$ might produce satisfactory results. The fact that the distance between their cylinder and the downstream boundary was only 8.5 diameters may be one reason why Thoman and Szewczyk had difficulty with other boundary conditions. The asymmetric program developed here extends to 50 diameters.

### 5.3.4 Changing regions

At the place where the mesh distribution across the rows changes, a special treatment of the data must take place. If more mesh points are added, an interpolation is required. This is the case in going from Region 2 to Region 3. It can be seen that the new points needed have been placed exactly halfway between the existing points so that a simple averaging can be used to obtain the new values.

The process is as follows. The last row in Region 2 is moved to a work array. A second work array is filled consecutively with values from the first array but every other position is left blank. From the values on each side of a blank an average is obtained and inserted in the blank space. This new work area is then moved to the first row in Region 3 and the calculations proceed as usual.

This method works even when simple averaging is not used. The appropriate number of blanks must be left in the new work area and interpolation on the bracketing values must then be carried out using the coordinate values of the points involved.

## 5.4 Current Status of Program

The overall computational scheme to be used in the solution of the asymmetric problem has been developed and can be seen in the structure of the procedures used in CICHY/UNS in Appendix E. Much of the detail discussed in the preceding sections has yet to be programmed. Keeping track of indicies and control modes represents a most difficult task but frequent reference to the allocation sketches has proved very helpful.

## 5.5 Conclusions

The asymmetric problem is an excellent test case for evaluating the use of ILLIAC IV in solving fluid dynamic problems. The features which make it most attractive are its large size, requiring the use of buffers and overlapping I/O, and its nonuniform mesh size, with the accompanying indexing problems. These features are held in common with many of the types of fluid dynamic problems invisioned for solution on ILLIAC IV. Thus if this problem can be solved efficiently and rapidly, there is great hope for the increased use of ILLIAC IV in numerical fluid dynamics.

The programming of a large problem such as this can be speeded by the use of storage allocation charts. These are particularly helpful in determining the correct indicies and enabling mode patterns.

The use of a buffer system to handle problems with large data sets appears to work well. Only when actual I/O times and processor times are available can the actual efficiency of this method be determined.

## 6. FUTURE PROGRAMS IN THIS PROJECT

There are several interesting studies which can be built on the present work. First, of course, the asymmetric program should be completed. This, along with the symmetric program, should then be run on ILLIAC IV and careful account made of the time actually needed to solve the problem. The resulting solutions should then be compared to the serial solutions previously obtained and some conclusions drawn as to the benefit in speed and accuracy realized from using ILLIAC IV.

The effect of the actual mesh point density and distribution could be carried out easily once the asymmetric program is completed. It would be interesting to determine how to construct a mesh with just enough points to resolve the fluid motion accurately.

The effect of several downstream boundary conditions on the numerical solution could also be evaluated using the asymmetric program. It would be particularly interesting to evaluate boundary layer type conditions, since they imply that any fluid motions beyond the downstream boundary will not feed back to affect the solution near the cylinder.

An entirely different numerical approach to the problem of flow around a cylinder might be tried. Recently finite element methods have emerged as possible ways to solve fluid flow problems. Adapting these techniques for use on ILLIAC IV could be a useful and rewarding endeavor.

## REFERENCES

[1] Denenberg, S. A., "An Introductory Description of the ILLIAC IV System," CAC Document No. 10, Center for Advanced Computation, University of Illinois at Urbana-Champaign.

[2] Ericksen, J., "Iterative and Direct Methods for Solving Poisson's Equation and their Adaptability to ILLIAC IV", CAC Document No. 60, (1972), Center for Advanced Computation, University of Illinois at Urbana-Champaign.

[3] Hockney, R. W., "The Potential Calculation and Some Applications", Methods in Computational Physics, Volume 9.

[4] Son, J. S. and Hanratty, T. J., "Numerical Solution for the Flow Around a Cylinder at Reynolds Numbers of 40, 200 and 500", J. of Fluid Mechanics 35 (2), 369 (1969).

[5] Thoman, D. C. and Szewczyk, A. A., "Numerical Solutions of Time Dependent Two Dimensional Flow of a Viscons, Incompressible Fluid Over Stationary and Rotating Cylinders", Technical Report #66-14 (1966), Heat Transfer and Fluid Mechanics Laboratory, Department of Mechanical Engineering, University of Notre Dame.

APPENDIX A

```
$ZIP                                                                    00000100
$LIBRARY                                                                00000200
                              % CICHY/GLYSOND                           00000300
                              %   5/23/72                               00000400
        BEGIN                                                           00000500
        CREAL PI, DETA, DTSI, DT, DX, IDX, RLXP, LIMIT, NREY, KE,       00000600
                      TLMT, DRE, ETAMAX, TSIMAX, TERM4I,TERMB, TERMB2,  00000700
                      TERMD,TERMD2,DTI2,AT,BT,FT,CT,ET,GT,TA,DXS;       00000800
        CINT  NETA, NTSI, LL1, LL2,LL3,LL4,LL5,LL6, NITER,PINC,LM ,     00000900
                      EVEN ;                                            00001000
        PINT  CEVEN, CODD;                                              00001100
        PREAL  ETA, M, S ;                                              00001200
        PREAL VECTOR PSI[ 20], ZETA[ 20] ;                              00001300
        BOOLEAN BCS1, BC11, BCS, SAVEMODE, BC, AD,BCT, BEV,BOD,BEVEN,   00001400
                      BODD;                                             00001500
        FILE LINE (10 ROWS ) ;                                          00001600
        FILE I4DISK1 ="A / DISKFILE"( 90 ROWS FULL ) ;                  00001700
        LABEL FINI ;                                                    00001800
 %                                                                      00001900
 %                                                                      00002000
 %                                                                      00002100
 %  ***************************************************************** 00002200
 %                                                                      00002300
                      PE REAL SUBROUTINE                                00002400
    EXP AS RGA (PE REAL ARG AS RGA);                                    00002500
                      % EXPONENTIAL                                     00002600
        BEGIN                                                           00002700
$*      CALL EXP64 ;                                                    00002800
        END ;                                                           00002900
 %  ***************************************************************** 00003000
 %                                                                      00003100
                      PE REAL SUBROUTINE                                00003200
    SIN AS RGA (PE REAL ARG AS RGA);                                    00003300
                      % SINE                                            00003400
        BEGIN                                                           00003500
$*      CALL SIN64 ;                                                    00003600
        END ;                                                           00003700
 %  ***************************************************************** 00003800
 %                                                                      00003900
                      PE REAL SUBROUTINE                                00004000
    LN AS RGA ( PE REAL ARG AS RGA ) ;                                  00004100
                      % NATURAL LOGARITHM                               00004200
        BEGIN                                                           00004300
$*      CALL LN64 ;                                                     00004400
        END ;                                                           00004500
 %  ***************************************************************** 00004600
 %                                                                      00004700
                      PE REAL SUBROUTINE                                00004800
    ARCTAN AS RGA (PE REAL ARG AS RGA) ;                                00004900
                      % ARCTANGENT                                      00005000
        BEGIN                                                           00005100
$*      CALL ARCTAN64 ;                                                 00005200
        END ;                                                           00005300
 %  ***************************************************************** 00005400
 %                                                                      00005500
                      PE REAL SUBROUTINE                                00005600
    PRINT1 (PCPOINT VAR, CINT PRNT) ;                                   00005700
                      % PRINTS VALUES OF VARIABLE VAR FOR               00005800
                      %      THE ENTIRE NET                             00005900
        BEGIN                                                           00006000
```

```
      CINT  KL ;                                                        00006100
                    % PARAMETERS REQUIRED ZETA[KL],LL6                  00006200
      SAVEMODE := MODE ;                                                00006300
      MODE := BCS ;                                                     00006400
      LOOP  KL := 0, 1, LL6  DO                                         00006500
           BEGIN                                                        00006600
           IF PRNT EQL 1 THEN SIMWRITE (LINE,                           00006700
                      "VORTICITY, ROW NUMBER ", KL ) ;                  00006800
           IF PRNT EQL 2 THEN SIMWRITE (LINE,                           00006900
                      "STREAMFUNCTION, ROW NUMBER ", KL ) ;             00007000
           IF PRNT EQL 3 THEN SIMWRITE(LINE,                            00007100
                      "INITIAL STREAMFUNCTION, ROW NUMBER ",KL);        00007200
           SIMWRITE (LINE, VAR[KL] ) ;                                  00007300
           END ;                                                        00007400
      MODE := SAVEMODE ;                                                00007500
      END ;                                                             00007600
%  ***************************************************************************00007700
%                                                                       00007800
                    PE REAL SUBROUTINE                                  00007900
SYSTEMPARAMETERS ;                                                      00008000
      BEGIN                                                             00008100
      MODE := TRUE ;                                                    00008200
      NREY := 40.0 ;                                                    00008300
      DT := 0.04   ;                                                    00008400
      TLMT := 0.08;                                                     00008500
      LIMIT := 0.00001     ;                                            00008600
      NETA := 9 ;                                                       00008700
                    % NUMBER OF THETA MESH POINTS                       00008800
      NTSI := 17 ;                                                      00008900
                    % NUMBER OF RADIAL MESH POINTS                      00009000
      ETAMAX := 1.0 ;                                                   00009100
      TSIMAX := 1.5 ;                                                   00009200
      PINC := 1 ;                                                       00009300
      END ;                                                             00009400
%  ***************************************************************************00009500
%                                                                       00009600
                    PE REAL SUBROUTINE                                  00009700
COMPUTESYSTEMCONSTANTS ;                                                00009800
      BEGIN                                                             00009900
      MODE := TRUE ;                                                    00010000
      PI := 4 * ARCTAN(1) ;                                             00010100
      BCS1 := BOOLEAN(8000000000000000(16) )  ;                         00010200
      BC11 := BOOLEAN(0C000000000000000(16) )  ;                        00010300
      AD :=  BOOLEAN(0000000000000001(16)) ;                            00010400
      DETA := ETAMAX /(NETA - 1) ;                                      00010500
      DTSI := TSIMAX /(NTSI - 1) ;                                      00010600
      LL1 := NTSI - 2;                                                  00010700
                    % LOOP LIMIT ONE, THE NUMBER OF * INTERNAL *       00010800
                    %    MESH POINTS IN THE RADIAL  (TSI)  DIRECTION00010900
      LL2 := NTSI - 3;                                                  00011000
                    % LOOP LIMIT TWO, INDICATES TO THE PROGRAM         00011100
                    %    WHEN THE LAST SECTION HAS BEEN PROCESSED,     00011200
                    %    OR,  WHEN THE LAST ROW HAS BEEN PROCESSED     00011300
                    %    IN THE THOMAS METHOD.                         00011400
      IF LL1 LEQ 62 THEN LL3 := LL1                                     00011500
                    ELSE BEGIN                                          00011600
                         IF LL1 LEQ 124 THEN LL3 := LL1/2               00011700
                    ELSE BEGIN                                          00011800
                         IF LL1 LEQ 186 THEN LL3 := LL1/3               00011900
                    ELSE BEGIN                                          00012000
```

```
                    IF LL1 LEQ 248 THEN LL3 := LL1/4                    00012100
               ELSE BEGIN  GO TO FINI END                               00012200
                    END  END  END ;                                     00012300
               % LOOP LIMIT THREE, THE NUMBER OF INTERNAL               00012400
               %    RADIAL MESH POINTS IN A SECTION, GIVEN              00012500
               %    HERE AT ITS MAXIMUM VALUE OF 62.  THIS              00012600
               %    NUMBER MUST BE DETERMINED BY THE USER TO            00012700
               %    FIT THE NET LENGTH SO THAT  LL1/LL3  IS             00012800
               %    A WHOLE NUMBER.                                     00012900
     LL4 := NETA - 2 ;                                                  00013000
               % LOOP LIMIT FOUR, INDICATES THE NUMBER OF               00013100
               %    INTERNAL MESH POINTS IN THE CIRCUMFERENTIAL         00013200
               %    (ETA) DIRECTION. THE MAXIMUM NUMBER IS 62           00013300
     LL5 := NETA - 3 ;                                                  00013400
               % LOOP LIMIT FIVE, INDICATES WHEN THE LAST               00013500
               %    COLUMN HAS BEEN PROCESSED IN THE                    00013600
               %    THOMAS METHOD                                       00013700
     LL6 := NTSI - 1 ;                                                  00013800
     KE := PI * DTSI ;                                                  00013900
     DX := EXP(KE) ;                                                    00014000
     IDX := 1.0/DX ;                                                    00014100
               %                                                        00014200
               % CALCULATE THE RELAXATION PARAMETER                     00014300
               %                                                        00014400
          BEGIN                                                         00014500
          CREAL AR, BR, CR, DR ;                                        00014600
          AR := 1.0 / ( NETA * NETA ) ;                                 00014700
          BR := 1.0 / ( NTSI * NTSI ) ;                                 00014800
          CR := (AR + BR ) * 0.5 ;                                      00014900
          DR := 0.5 * LN(CR) ;                                          00015000
          DRE := EXP(DR) ;                                              00015100
          RLXP := 2.0 / (1.0 + PI * DRE ) ;                             00015200
RLXP := 1.84 ;                                                          00015300
          END ;                                                         00015400
               %                                                        00015500
               % CALCULATE THE BOOLEAN MASKS NEEDED FOR THE            00015600
               %    ARRAY WIDTH USED.                                   00015700
               %                                                        00015800
          BEGIN                                                         00015900
          CINT J ;                                                      00016000
          BC := FALSE ;                                                 00016100
          LOOP J:=1,1,LL4   DO                                          00016200
               BEGIN                                                    00016300
               BC := BC OR BCS1 ;                                       00016400
               BC := REVR(1,BC) ;                                       00016500
               END ;                                                    00016600
          BCS := REVR(1,BC) ;                                           00016700
          BCS := BCS OR BC11 ;                                          00016800
          END ;                                                         00016900
               % THE BOOLEAN VARIABLE BC MUST REFLECT THE               00017000
               %    WIDTH OF THE NET.  A "1" SHOULD BE PLACED           00017100
               %    IN EACH BIT POSITION WHICH CORRESPONDS              00017200
               %    TO A PE CONTAINING ONE OF THE INTERNAL MESH         00017300
               %    POINTS. ALL OTHER BITS MUST BE ZERO.                00017400
          BEGIN                                                         00017500
          CINT J;                                                       00017600
          BCT := FALSE ;                                                00017700
          LOOP J := 1,1,LL3 DO                                          00017800
              BEGIN                                                     00017900
              BCT := BCT OR BCS1 ;                                      00018000
```

```
                BCT := REVR(1,BCT);                                    00018100
                END;                                                   00018200
           END;                                                        00018300
      TERM4I := 0.25 /(DTSI * DETA) ;                                  00018400
      TERMB  := 2.0  /(NREY * DETA * DETA) ;                           00018500
      TERMB2 :=  2.0 * TERMB ;                                         00018600
      TERMD  := 2.0  /(NREY * DTSI * DTSI) ;                           00018700
      TERMD2 :=  2.0 * TERMD ;                                         00018800
      DTI2   :=  2.0/DT ;                                              00018900
%                                                                      00019000
      BEVEN := BOOLEAN(2AAAAAAAAAAAAAAA(16) ) ;                        00019100
      BODD := BOOLEAN(5555555555555554(16) ) ;                         00019200
      IF ( ( LL1 DIV 2) * 2 - LL1 ) = 0                                00019300
                     THEN BEGIN                                        00019400
                          EVEN := 1 ;                                  00019500
                          LM := LL1 ;                                  00019600
                          END                                          00019700
                     ELSE BEGIN                                        00019800
                          EVEN := 0 ;                                  00019900
                          LM := LL2 ;                                  00020000
                          END ;                                        00020100
      BOD := BODD AND BC ;                                             00020200
      BEV := BEVEN AND BC ;                                            00020300
      CEVEN := 0 ;                                                     00020400
      CODD := 0 ;                                                      00020500
      DXS := DX * DX ;                                                 00020600
      MODE := BEV ;                                                    00020700
      CEVEN := 1 ;                                                     00020800
      MODE := TRUE ;                                                   00020900
      MODE := BOD ;                                                    00021000
      CODD := 1 ;                                                      00021100
%                                                                      00021200
      AT    := RLXP * DTSI * DTSI ;                                    00021300
      BT    := 2.0 * (DTSI * DTSI + DETA * DETA) ;                     00021400
      CT := AT/BT ;                                                    00021500
      FT := RLXP * DETA * DETA ;                                       00021600
      ET := FT/BT ;                                                    00021700
      GT := CT * DETA * DETA ;                                         00021800
      TA := 1.0 - RLXP ;                                               00021900
%                                                                      00022000
      MODE := TRUE ;
      MODE := (PEN LEQ (NETA - 1) / 2 ) ;
      ETA := PEN ;
      MODE := TRUE ;
      MODE := (PEN GTR (NETA - 1) / 2 ) ;
      ETA := (NETA - 1 ) - PEN ;
      MODE := BC ;                                                     00022100
      M := ETA * PI * DETA ;
      S := SIN(M) ;                                                    00022400
      MODE := TRUE ;                                                   00022500
      END ;                                                            00022600
%  ************************************************************************00022700
%                                                                      00022800
                     PE REAL SUBROUTINE                                00022900
  COMPUTEADICOEFFICIENTSANDSKEW (CREAL E, CNPOINT TERME,               00023000
                     CNPOINT COFA, PCPOINT KFA, PCPOINT KFB,           00023100
                     PCPOINT KFC, PCPOINT KFD ) ;                      00023200
      BEGIN          % SKEW COEFFICIENTS                               00023300
      CINT K,J,L ;                                                     00023400
      PREAL TERMA, TERMC ;                                             00023500
```

```
      BOOLEAN BCI ;                                                    00023600
      MODE := BC ;                                                     00023700
                     % IF THE NET LENGTH (RADIAL DIRECTION) IS         00023800
                     %    LONGER THAN 62, THE NET MUST BE SEPARATED   00023900
                     %    INTO SEVERAL EQUAL LENGTH SECTIONS. A       00024000
                     %    SECTION MAY AT MOST BE 62 ROWS LONG         00024100
                     %    (LL3=62) AND 62 WIDE (LL4=62) .  THE        00024200
                     %    PROGRAM TREATS ONE SECTION AT A TIME UNTIL  00024300
                     %    ALL SECTIONS HAVE BEEN SOLVED FOR THE       00024400
                     %    NEW VORTICITY VALUES.  LL2 TELLS THE        00024500
                     %    PROGRAM WHEN THE LAST SECTION HAS BEEN      00024600
                     %    PROCESSED.                                  00024700
      LOOP  K := 0,LL3, LL2   DO                                       00024800
           BEGIN                                                       00024900
           BCI := BC ;                                                 00025000
           MODE := BCI ;                                               00025100
           LOOP  J := 1, 1, LL3   DO                                   00025200
                     % THERE ARE LL3 ROUTES IN EACH                    00025300
                          %    SECTION TO SKEW THE ARRAY.             00025400
                BEGIN                                                  00025500
                L := K + J  ;                                          00025600
                TERMA :=-(PSI[L+1] -PSI[L-1]) * TERM4I ;               00025700
                TERMC :=-(RTL(1,TRUE,PSI[L])-RTR(1,TRUE,PSI[L]))       00025800
                     * TERM4I ;                                        00025900
                TERME[L] := E*E*DTI2 ;                                 00026000
                COFA[L] := TERME[L] + TERMB2 ;                         00026100
                     % " A "                                           00026200
                E := E * DX ;                                          00026300
                KFA[L]  := RTR(J-1,    ,TERMA - TERMB  );              00026400
                     % " C "                                           00026500
                KFB[L] := RTR(J-1,     ,-TERMA - TERMB ) ;             00026600
                     % " B "                                           00026700
                KFC[L]:= RTR(J-1,     ,(TERMC+TERMD) * ZETA[L-1]       00026800
                     +(TERME[L]-TERMD2) * ZETA[L]                      00026900
                     +(-TERMC+TERMD) * ZETA[L+1]);                     00027000
                     % " D "                                           00027100
                KFD[L] := COFA[L] ;                                    00027200
                BCI := REVR(1,BCI) ;                                   00027300
                MODE := BCI ;                                          00027400
                END ;                                                  00027500
           END ;                                                       00027600
      MODE := TRUE ;                                                   00027700
      END ;                                                            00027800
%  ***************************************************************** 00027900
%                                                                     00028000
                     PE REAL SUBROUTINE                                00028100
THOMASONROWS (PCPOINT KFA, PCPOINT KFB, PCPOINT KFC, PCPOINT KFD ) ;  00028200
      BEGIN          % APPLY THE THOMAS METHOD TO EACH ROW IN THE      00028300
                     %    SECTION SIMULTANEOUSLY THUS INVERTING        00028400
                     %    LL3 MATICIES AT A TIME.                      00028500
                     %                                                 00028600
                     % REFERENCE TEXT: LAPIDUS, L. , DIGITAL          00028700
                     %    COMPUTATION FOR CHEMICAL ENGINEERS,          00028800
                     %    MC GRAW-HILL, 1962, P.254.                   00028900
      CINT J,K ;                                                       00029000
      MODE := TRUE ;                                                   00029100
      LOOP  K := 0,LL3, LL2   DO                                       00029200
           BEGIN                                                       00029300
           PINT  R ;                                                   00029400
           BOOLEAN BCI ;                                               00029500
```

```
      BEGIN                  % WORKING FORWARD COMPUTE          00029600
                             %    INTERMEDIATE VALUES           00029700
      CINT  N ;                                                 00029800
                             % BC CARRIES THE BIT PATTERN       00029900
                             %    REFLECTING THE WIDTH OF THE   00030000
                             %    NET. THIS INFORMATION IS      00030100
                             %    PASSED TO BCI FOR USE IN      00030200
                             %    ENABLING THE PROPER           00030300
                             %    PROCESSING ELEMENTS WHILE     00030400
                             %    OPERATING ON THE SKEWED       00030500
                             %    ARRAY.                        00030600
      BCI := BCT;                                               00030700
      MODE := TRUE ;                                            00030800
      R :=  PEN + K ;                                           00030900
                             % R IS LIKE A POINTER WHICH        00031000
                             %    POINTS TO THE PROPER          00031100
                             %    VECTOR ELEMENT IN EACH PE     00031200
      MODE := BCI ;                                             00031300
                             % W(1) = A(1)                      00031400
      KFB[R] := KFB[R] / KFD[R] ;                               00031500
                             % Q(1) = B(1) / W(1)               00031600
      KFC[R] := KFC[R] / KFD[R] ;                               00031700
                             % G(1) = D(1) / W(1)               00031800
      LOOP  J := 2,1,LL4 DO                                     00031900
           BEGIN                                                00032000
           MODE := TRUE ;                                       00032100
                             % SHIFT INDEXING AND ENABLING      00032200
                             %    PARAMETERS                    00032300
           R  := RTR(1,MODE,R) ;                                00032400
           BCI:= REVR(1,BCI)   ;                                00032500
           MODE := BCI ;                                        00032600
           KFD[R]:=  KFD[R]  - KFA[R]                           00032700
                             *RTR(1,  ,KFB[R+1]) ;              00032800
                             % W(R) = A(R) - C(R) * Q(R-1)      00032900
           KFB[R] := KFB[R] / KFD[R] ;                          00033000
                             % Q(R) = B(R) / W(R)               00033100
           KFC[R]:=(KFC[R]- KFA[R]                              00033200
                             * RTR(1,  ,KFC[R+1])   )           00033300
                             / KFD[R] ;                         00033400
                             % G(R)=( D(R)-C(R)*G(R-1) )/W(R)   00033500
           END ;                                                00033600
      END ;                                                     00033700
      BEGIN                  % WORK BACKWARD TO GET SOLUTION    00033800
                             %    VECTORS                       00033900
      KFD[R] := KFC[R] ;                                        00034000
                             % X(N) = G(N)                      00034100
      LOOP   J := 1,1,LL5    DO                                 00034200
           BEGIN                                                00034300
           MODE := TRUE ;                                       00034400
           R := RTL(1,MODE,R) ;                                 00034500
           BCI := REVL(1,BCI) ;                                 00034600
           MODE := BCI ;                                        00034700
           KFD[R] := KFC[R]-KFB[R]                              00034800
                             *RTL(1,  ,KFD[R-1]) ;              00034900
                             % X(R) = G(R) - Q(R) * X(R+1)      00035000
                             % NEW VORTICITY VALUES STORED     00035100
                             %    IN SKEWED FORM IN KFD[R]      00035200
           END ;                                                00035300
      END ;                                                     00035400
END ;                                                           00035500
```

```
        MODE := TRUE ;
        END :
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    UNSKEWANDSTOREINTERMEDIATEVORTICITY (PCPOINT KFD) ;
        BEGIN           % UNSKEW RESULTS AND OVERLAY VORTICITY NET
                        %     WITH THE NEW VALUES
        CINT J,K,L ;
        MODE := BC ;
        LOOP  K := 0, LL3,LL2   DO
            BEGIN
            LOOP  J := 1,1,LL3  DO
                BEGIN
                L := K + J ;
                ZETA[L] := RTL(J-1,TRUE,KFD[L]) ;
                END ;
            END ;
        MODE := TRUE ;
        END ;
                        %  W[R] ::= KFD[R]      OR  D2
                        %  Q[R] ::= KFB[R]  OR  B
                        %  G[R] ::= KFC[R]  OR  D1
                        %  X[R] ::= KFD[R]  OR  D2,W[R]
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    COMPUTENEWADICOEFFICIENTS (CNPOINT TERME, CNPOINT COFA,
                        PCPOINT KFA, PCPOINT KFB, PCPOINT KFC,
                        PCPOINT KFD ) ;
        BEGIN           % DEVELOP NEW COEFFICIENTS
        CINT J,K,L ;
        PREAL TERMA, TERMC ;
        MODE := BC ;
        LOOP  J := 1, 1, LL1  DO
            BEGIN
            TERMA :=-(PSI[J+1] -PSI[J-1]) * TERM4I ;
            TERMC :=-(RTL(1,TRUE,PSI[J]) - RTR(1,TRUE,PSI[J]))
                        * TERM4I ;
            COFA[J] := TERME[J] + TERMD2 ;
                        % "A"
            KFA[J]  := (-TERMA+TERMB)* RTR(1,TRUE,ZETA[J])
                        +(TERME[J] - TERMB2 ) * ZETA[J]
                        +(TERMA+TERMB) * RTL(1,TRUE,ZETA[J])  ;
                        % "D"
            KFB[J]:= (TERMC-TERMD) ;
                        % "B"
            KFC[J]:= (-TERMC-TERMD) ;
                        % "C"
            END ;
        MODE := TRUE ;
        END ;
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    THOMASONCOLUMNSANDSTOREUPDATEDVORTICITY (CNPOINT TERME,
                        CNPOINT COFA, PCPOINT KFA, PCPOINT KFB,
                        PCPOINT KFC, PCPOINT KFD ) ;
        BEGIN
        CINT L,J ;
```

```
          BEGIN     % THOMAS METHOD ON EACH COLUMN                        00041600
          MODE := BC ;                                                    00041700
          KFD[1]   := COFA[1] ;                                           00041800
                    % W(1) = A(1), (747+1)                                00041900
          KFB[1]   := KFB[1]   / KFD[1] ;                                 00042000
                    % Q(1) = B(1) / W(1), (249+1), (747+1)                00042100
          KFA[1] := ( KFA[1] - KFC[1] * ZETA[0] ) / KFD[1] ;              00042200
                    % G(1) = D(1) / W(1), (747+1)                         00042300
          LOOP  J := 2, 1, LL1  DO                                        00042400
               BEGIN                                                      00042500
               KFD[J]:=COFA[J]-KFC[J]*KFB[J-1] ;                          00042600
                    % W(R) = A(R) - C(R) * Q(R-1) , (249-1)               00042700
               KFB[J]:=KFB[J] / KFD[J] ;                                  00042800
                    % Q(R) = B(R) / W(R)                                  00042900
               KFA[J]:=(KFA[J]-KFC[J]*KFA[J-1])/KFD[J] ;                  00043000
                    % G(R) = (D(R) - C(R) * G(R-1) ) / W(R)               00043100
               END ;                                                      00043200
          END ;                                                           00043300
          BEGIN     % ENTER RESULTS IN VORTICITY NET                     00043400
          ZETA[LL1] := KFA[LL1] ;                                         00043500
                    % X(N) = G(N)                                        00043600
          LOOP  J := 1, 1, LL2  DO                                        00043700
               BEGIN                                                      00043800
               L := LL1 - J ;                                             00043900
               ZETA[L] := KFA[L] - KFB[L] * ZETA[L+1] ;                   00044000
                    % X(R) = G(R) - Q(R) * X(R+1),  (747+1)              00044100
               END ;                                                      00044200
          MODE := TRUE ;                                                  00044300
$*        DISPLAY 0,1;                                                    00044400
          END ;                                                           00044500
     END ;                                                                00044600
% ***************************************************************************00044700
%                                                                         00044800
                    PE REAL SUBROUTINE                                    00044900
VORTICITY ( PCPOINT PSI, PCPOINT ZETA ) ;                                 00045000
     BEGIN          % COMPUTE NEW VALUES OF THE VORTICITY USING          00045100
                    %    THE PEACEMAN RACHFORD ALTERNATING DIRECTION     00045200
                    %    IMPLICIT METHOD  ( ADI )                        00045300
     CREAL E ;                                                            00045400
     PREAL VECTOR KFA[ 20],KFB[ 20],KFC[ 20],KFD[ 20];                    00045500
     CREAL VECTOR TERME[ 20] , COFA[ 20] ;                                00045600
                    % PARAMETERS REQUIRED, DTSI, DETA, NREY, PI, DX,     00045700
                    %    DT, BC, LL1,LL2,LL3,LL4,LL5,BCI,                00045800
                    %    TERM4I,TERMB,TERMB2,TERMD,TERMD2,               00045900
                    %    DTI2,BCT                                        00046000
                                                                          00046100
                    %                                                   00046200
     MODE := BC ;                                                         00046300
     E := PI * DX ;                                                       00046400
$*   DISPLAY 0,1;                                                         00046500
                    % IN THE FIRST HALF TIME STEP THE VORTICITY          00046600
                    %    IS IMPLICIT ACROSS THE ROWS. BY SKEWING         00046700
                    %    THE ARRAYS OF COEFFICIENTS, ALL ROWS CAN        00046800
                    %    BE OPERATED ON SIMULTANEOUSLY. THE              00046900
                    %    THOMAS METHOD IS APPLIED SEQUENTIALLY TO        00047000
                    %    EACH ELEMENT IN A ROW.                          00047100
                    %                                                   00047200
     MODE := TRUE ;                                                       00047300
```

```
      COMPUTEADICOEFFICIENTSANDSKEW (E,TERME,COFA,KFA,KFB,KFC,KFD) ;
      THOMASONROWS (KFA,KFB,KFC,KFD) ;
      UNSKEWANDSTOREINTERMEDIATEVORTICITY (KFD) ;
                        %
                        % IN THE SECOND HALF TIME STEP THE VORTICITY
                        %    IS IMPLICIT WITHIN THE COLUMNS.  WITH
                        %    THE COEFFICIENTS STORED STRAIGHT, ALL
                        %    COLUMNS CAN BE OPERATED ON SIMULTANEOUSLY.
                        %    THE THOMAS METHOD IS APPLIED SEQUENTIALLY
                        %    TO EACH ELEMENT IN A COLUMN.
                        %
      COMPUTENEWADICOEFFICIENTS (TERME,COFA,KFA,KFB,KFC,KFD) ;
      THOMASONCOLUMNSANDSTOREUPDATEDVORTICITY (TERME,COFA,
                        KFA,KFB,KFC,KFD) ;
      MODE := TRUE ;
      END ;
%  ************************************************************************
%
                        PE REAL SUBROUTINE
    NEWPSI (PINT J,PINT K,CINT OUT LIM,PREAL OUT E,BOOLEAN PMODE) ;
      BEGIN
      PREAL TRMA,TRMB,TRMC,TRMD,TEST,DIF ;
      MODE := PMODE ;
      TRMA := TA * PSI[J] ;
      TRMB := CT * (RTL(1,,PSI[K] )
                        + RTR(1,,PSI[K])) ;
      TRMC := ET *(PSI[J+1] + PSI[J-1]) ;
      TRMD :=( E * E * ZETA[J]) * GT ;
                        %
                        % SAVE THE OLD VALUE OF THE STREAM FUNCTION.
                        %
      TEST := PSI[J] ;
                        %
                        % COMPUTE NEW VALUES OF STREAM FUNCTION.
                        %
      PSI[J] := TRMA + TRMB + TRMC - TRMD ;
                        %
                        % TEST FOR CONVERGENCE
                        %
      DIF  := PSI[J] - TEST ;
      IF ABS(DIF) GTR  LIMIT  THEN  LIM := 1 ;
      E := E * DXS ;
      END ;
%  ************************************************************************
%
                        PE REAL SUBROUTINE
    STREAMFUNCTION ( PCPOINT PSI, PCPOINT ZETA ) ;
      BEGIN             % DETERMINE THE NEW VALUES OF THE STREAM
                        %     FUNCTION USING THE MODIFIED SUCCESSIVE
                        %     OVERRELAXATION (MSOR) METHOD
      LABEL ITERATE ;
      PREAL  E ;
      CINT C, PASS, LIM ;
      PINT J, K, SW1, SW2 ;
      BOOLEAN   BEND, PMODE ;
                        % PARAMETERS REQUIRED, RLXP, DX, BC, DTSI, DETA,
                        %    PI, LL1, LL2, LIMIT,AT,BT,CT,
                        %    ET,FT,GT,TA,DXS,PINC,LM
      MODE := BC ;
      NITER := 0 ;
```

```
ITERATE:    BEGIN                                                          00053400
            LABEL CONVERGED, PASS2 ;                                       00053500
                      % THE MODIFIED SUCCESSIVE OVERRELAXATION METHOD      00053600
                      %    IS AN ITERATIVE METHOD WHICH REQUIRES THAT      00053700
                      %    SUCCESSIVE VALUES OF THE STREAM FUNCTION        00053800
                      %    CONVERGE TO THE TRUE VALUE.  CONVERGENCE        00053900
                      %    IS MEASURED BY COMPUTING THE ABSOLUTE           00054000
                      %    DIFFERENCE BETWEEN THE VALUE OF THE STREAM      00054100
                      %    FUNCTION ON TWO SUCCESSIVE ITERATES             00054200
                      %    AND COMPARING IT TO AN ERROR                    00054300
                      %    LIMIT.   WHEN THE ERROR IS                      00054400
                      %    LESS THAN THE LIMIT AT EACH MESH POINT,         00054500
                      %    CONVERGENCE IS ASSUMED.  LIM IS A CU            00054600
                      %    VARIABLE WHICH IS SET TO ZERO AT THE            00054700
                      %    BEGINNING OF EACH ITERATION.  IF ANY MESH       00054800
                      %    POINT FAILS THE CONVERGENCE TEST, LIM           00054900
                      %    IS THEN SET EQUAL TO ONE AND THUS               00055000
                      %    INDICATES TO THE PROGRAM THAT ANOTHER           00055100
                      %    ITERATION IS NEEDED.                            00055200
            LIM := 0 ;                                                     00055300
                      % DEVELOP THE TERMS IN THE FINITE DIFFERENCE         00055400
                      %    FORMULA FOR FINDING AN IMPROVED VALUE           00055500
                      %    OF THE STREAM FUNCTION.                         00055600
$*          DISPLAY 0,1;                                                   00055700
            PASS := 1 ;                                                    00055800
            SW1 := CEVEN ;                                                 00055900
            SW2 := CODD ;                                                  00056000
            BEND := BOD ;                                                  00056100
            NITER := NITER + 1 ;                                           00056200
PASS2:      FOR ALL SW1 = 1 DO E := PI * DXS ;                             00056300
            FOR ALL SW1 = 0 DO E := PI * DX ;                              00056400
            LOOP C := 1, 2, LM DO                                          00056500
                 BEGIN                                                     00056600
                 J := C + SW1 ;                                            00056700
                 K := C + SW2 ;                                            00056800
                 PMODE := MODE ;                                           00056900
                 NEWPSI (J,K,LIM,E,PMODE) ;                                00057000
                 MODE := BC ;                                              00057100
                 END ;                                                     00057200
                      % IF THERE ARE AN ODD NUMBER OF ROWS IN THE          00057300
                      %    PSI NET, THE LAST ROW MUST BE UPDATED BY        00057400
                      %    THE FOLLOWING PROCEDURE WHICH OPERATES          00057500
                      %    ON ONLY THE ODD COLUMNS IN PASS ONE AND         00057600
            IF EVEN = 0 THEN                                               00057800
                      %    THE EVEN COLUMNS IN PASS TWO.                   00057700
                 BEGIN                                                     00057900
                 J := LL1 ;                                                00058000
                 K := LL1 ;                                                00058100
                 PMODE := BEND ;                                           00058200
                 NEWPSI (J,K,LIM,E,PMODE) ;                                00058300
                 MODE := BC ;                                              00058400
                 END ;                                                     00058500
                      % AFTER PASS ONE THE CONTROL VARIABLES ARE           00058600
                      %    ADJUSTED FOR PASS TWO.                          00058700
            IF PASS = 1 THEN                                               00058800
                 BEGIN                                                     00058900
                 PASS := 2 ;                                               00059000
                 SW1 := CODD ;                                             00059100
                 SW2 := CEVEN ;                                            00059200
                 BEND := BEV ;                                             00059300
```

```
                GO TO PASS2 ;                                                   00059400
                END ;                                                           00059500
            IF LIM = 0 THEN GO TO CONVERGED ;                                   00059600
  IF NITER = 6 THEN GO TO CONVERGED ;                                           00059700
            GO TO ITERATE ;                                                     00059800
 CONVERGED: MODE := TRUE ;                                                      00059900
            END ;                                                               00060000
       END ;                                                                    00060100
%  ***************************************************************************00060200
%                                                                               00060300
                       PE REAL SUBROUTINE                                       00060400
   SURFACEVORTICITY (PCPOINT PSI, PCPOINT ZETA) ;                               00060500
                       % COMPUTE THE SURFACE VORTICIY                           00060600
                       %     USING A TAYLOR SERIES APPROXIMATION                00060700
       BEGIN                                                                    00060800
       CREAL PRODUCT, PREFIX ;                                                  00060900
                       % PARAMETERS REQUIRED, DTSI, PI                          00061000
       MODE := BCS ;                                                            00061100
       PRODUCT := PI * DTSI ;                                                   00061200
       PREFIX := 1.0 / ( 2.0 * PRODUCT * PRODUCT ) ;                            00061300
       ZETA[0] := PREFIX * ( 8.0 * PSI[1] - PSI[2] ) ;                          00061400
       MODE := TRUE ;                                                           00061500
       END ;                                                                    00061600
%  ***************************************************************************00061700
%                                                                               00061800
                       PE REAL SUBROUTINE                                       00061900
   INITIALIZE ( PCPOINT PSI, PCPOINT ZETA ) ;                                   00062000
       BEGIN                                                                    00062100
       CREAL EX, IX ;                                                           00062200
       CINT  JD, JDE ;                                                          00062300
                       % PARAMETERS REQUIRED, DX, IDX, LL1, S                   00062400
                       %                                                        00062500
            BEGIN      % USE THE POTENTIAL FLOW SOLUTION AS THE                 00062600
                       %    INITIAL STREAM FUNTION VALUES AND                   00062700
                       %    ZERO THE VORTICITY NET                              00062800
            MODE := BCS ;                                                       00062900
            PSI[0] := 0.0 ;                                                     00063000
            ZETA[0] := 0.0 ;                                                    00063100
            EX := DX ;                                                          00063200
            IX := IDX ;                                                         00063300
            LOOP  JD  := 1,1, LL1   DO                                          00063400
                 BEGIN                                                          00063500
                 MODE := BCS ;                                                  00063600
                 ZETA[JD] := 0.0 ;                                              00063700
                 PSI[JD] := 0.0 ;                                               00063800
                 MODE := BC ;                                                   00063900
                 PSI[JD]  := (EX - IX)*S ;                                      00064000
                 EX:= EX * DX ;                                                 00064100
                 IX:= IX * IDX ;                                                00064200
                 JDE  := JD ;                                                   00064300
                 END ;                                                          00064400
            JD := JDE + 1 ;                                                     00064500
            PSI[JD] := EX * S ;                                                 00064600
            ZETA[JD] := 0.0 ;                                                   00064700
            MODE := TRUE ;                                                      00064800
            END ;                                                               00064900
       PRINT1 (PSI,3) ;                                                         00065000
       STREAMFUNCTION ( PSI, ZETA ) ;                                           00065100
       SURFACEVORTICITY ( PSI, ZETA ) ;                                         00065200
       END;                                                                     00065300
```

```
%  ***************************************************************************00065400
%                                                                              00065500
                        PE REAL SUBROUTINE                                    00065600
    PRINTPARAMETERS ;                                                         00065700
         BEGIN                                                                00065800
         MODE := TRUE ;                                                       00065900
         SIMWRITE (LINE,"REYNOLDS NUMBER", NREY,                              00066000
                        "NUMBER OF CIRCUMFERENTIAL MESH POINTS", NETA,        00066100
                        "NUMBER OF RADIAL MESH POINTS",NTSI,                  00066200
                        "CIRCUMFERENTIAL INCREMENT SIZE",DETA,                00066300
                        "RADIAL INCREMENT SIZE",DTSI,                         00066400
                        "MAXIMUM CIRCUMFERENTIAL COORDINATE",ETAMAX,          00066500
                        "MAXIMUM RADIAL COORDINATE",TSIMAX,                   00066600
                        "DIMENTIONLESS TIME INCREMENT",DT,                    00066700
                        "MAXIMUM DIMENTIONLESS TIME DESIRED",                 00066800
                        TLMT, "NUMBER OF TIME STEPS BETWEEN PRINT-OUTS",      00066900
                        PINC,"RELAXATION PARAMETER",RLXP,                     00067000
                        "ERROR LIMIT FOR ITERATIVE CONVERGENCE",              00067100
                        LIMIT,"BOOLEAN MASKS FOR FIELD WIDTH",                00067200
                        BCS,BCT) ;                                            00067300
         END ;                                                                00067400
%  ***************************************************************************00067500
%                                                                              00067600
                        PE REAL SUBROUTINE                                    00067700
    OUTPUTRESULTS (CINT OUT ST, CREAL T) ;                                    00067800
         BEGIN                                                                00067900
         MODE := TRUE  ;                                                      00068000
         IF ST EQL PINC THEN                                                  00068100
              BEGIN                                                           00068200
              SIMWRITE(LINE,"DIMENSIONLESS TIME",T ) ;                        00068300
              PRINT1 (ZETA,1) ;                                               00068400
              PRINT1 (PSI,2) ;                                                00068500
              SIMWRITE (LINE, "NUMBER OF ITERATIONS = ", NITER );             00068600
              ST := 0 ;                                                       00068700
$*            DISPLAY 0,1;                                                    00068800
              END ;                                                           00068900
         ST := ST + 1 ;                                                       00069000
         END ;                                                                00069100
%  ***************************************************************************00069200
%                                                                              00069300
                        PE REAL SUBROUTINE                                    00069400
    MAIN ;                                                                    00069500
         BEGIN                                                                00069600
         LABEL REPEAT ;                                                       00069700
         CREAL  T ;                                                           00069800
                        % ACCUMULATED DIMENSIONLESS TIME                      00069900
         CINT ST ;                                                            00070000
                        % PRINTOUT PARAMETER                                  00070100
         MODE := TRUE ;                                                       00070200
         T := 0.0 ;                                                           00070300
         ST := 1 ;                                                            00070400
         SYSTEMPARAMETERS ;                                                   00070500
         COMPUTESYSTEMCONSTANTS ;                                             00070600
         PRINTPARAMETERS ;                                                    00070700
         INITIALIZE ( PSI, ZETA ) ;                                           00070800
         OUTPUTRESULTS (ST,T) ;                                               00070900
                        % INITIALIZES THE STREAM FUNCTION AND VORTICITY       00071000
                        %    NETS, RELAXES THE PSI NET, AND COMPUTES AN       00071100
                        %    INITIAL SURFACE VORTICITY                        00071200
  REPEAT:T := T + DT ;                                                        00071300
```

```
                    % INCREMENT TIME                                        00071400
        VORTICITY ( PSI, ZETA ) ;                                           00071500
                    % COMPUTE NEW VALUES OF VORTICITY BY ADI                 00071600
        STREAMFUNCTION ( PSI, ZETA ) ;                                      00071700
                    % COMPUTE NEW VALUES OF THE STREAM FUNCTION              00071800
                    %    BY THE MODIFIED SUCCESSIVE OVERRELAXATION          00071900
                    %    METHOD                                             00072000
        IF ST NEQ PINC THEN                                                 00072100
             SIMWRITE (LINE, "NUMBER OF ITERATIONS = ", NITER );            00072200
        SURFACEVORTICITY ( PSI, ZETA ) ;                                    00072300
                    % ESTIMATES SURFACE VORTICITY BY REFERENCE TO           00072400
                    %    THE STREAM FUNCTION VALUES NEAR THE SURFACE        00072500
        OUTPUTRESULTS (ST,T) ;                                              00072600
        IF T LSS TLMT  THEN   GO TO REPEAT ;                                00072700
        END ;                                                               00072800
%  ***************************************************************************00072900
%  ***************************************************************************00073000
%                                                                           00073100
        MAIN ;                                                              00073200
 FINI:  END .                                                               00073300
```

# APPENDIX B

```
$ZIP                                                                    00000200
$LIBRARY                                                                00000300
                          % CICHY/GLYDIRECT                             00000400
                          %   5/24/72                                   00000500
        BEGIN                                                           00000600
        CREAL PI, DETA, DTSI, DT, DX, IDX, NREY, KE,                    00000700
                  TLMT, ETAMAX, TSIMAX, TERM4I, TERMB, TERMB2,          00000800
                  TERMD,TERMD2,DTI2,HX,HY ;                             00000900
        CINT  NETA, NTSI, LL1, LL2,LL3,LL4,LL5,LL6, NITER,PINC,         00001000
                  CN,CM,PP,I,J,CK,CS,OFF,PPP,K ;                        00001100
        PINT  CEVEN, CODD;                                              00001200
        PINT VECTOR  INDEX[1] ;                                         00001300
        PREAL  ETA, M, S, X, E ;                                        00001400
        PREAL VECTOR PSI[ 20], ZETA[ 20],A[1],SS[1],IS[1] ;             00001500
        BOOLEAN BCS1, BC11, BCS, SAVEMODE, BC, AD,BCT ;                 00001600
        FILE LINE (10 ROWS ) ;                                          00001700
        FILE I4DISK1 ="A / DISKFILE"( 90 ROWS FULL ) ;                  00001800
        LABEL FINI ;                                                    00001900
%                                                                       00002000
%                                                                       00002100
%                                                                       00002200
% ***************************************************************************00002300
%                                                                       00002400
                          PE REAL SUBROUTINE                            00002500
    EXP AS RGA (PE REAL ARG AS RGA);                                    00002600
                          % EXPONENTIAL                                 00002700
        BEGIN                                                           00002800
$*      CALL EXP64 ;                                                    00002900
        END ;                                                           00003000
% ***************************************************************************00003100
%                                                                       00003200
                          PE REAL SUBROUTINE                            00003300
    SIN AS RGA (PE REAL ARG AS RGA);                                    00003400
                          % SINE                                        00003500
        BEGIN                                                           00003600
$*      CALL SIN64 ;                                                    00003700
        END ;                                                           00003800
% ***************************************************************************00003900
%                                                                       00004000
                          PE REAL SUBROUTINE                            00004100
    COS AS RGA (PE REAL ARG AS RGA);                                    00004200
                          % COSINE                                      00004300
        BEGIN                                                           00004400
$*      CALL COS64 ;                                                    00004500
        END ;                                                           00004600
% ***************************************************************************00004700
%                                                                       00004800
                          PE REAL SUBROUTINE                            00004900
    LN AS RGA ( PE REAL ARG AS RGA ) ;                                  00005000
                          % NATURAL LOGARITHM                           00005100
        BEGIN                                                           00005200
$*      CALL LN64 ;                                                     00005300
        END ;                                                           00005400
% ***************************************************************************00005500
%                                                                       00005600
                          PE REAL SUBROUTINE                            00005700
    ARCTAN AS RGA (PE REAL ARG AS RGA) ;                                00005800
                          % ARCTANGENT                                  00005900
        BEGIN                                                           00006000
$*      CALL ARCTAN64 ;                                                 00006100
```

```
        END ;
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    PRINT1 (PCPOINT VAR, CINT PRNT) ;
                        % PRINTS VALUES OF VARIABLE VAR FOR
                        %     THE ENTIRE NET
        BEGIN
        CINT  KL ;
                        % PARAMETERS REQUIRED ZETA[KL],LL6
        SAVEMODE := MODE ;
        MODE := BCS ;
        LOOP  KL := 0, 1, LL6  DO
             BEGIN
             IF PRNT EQL 1 THEN SIMWRITE (LINE,
                         "VORTICITY, ROW NUMBER ", KL ) ;
             IF PRNT EQL 2 THEN SIMWRITE (LINE,
                         "STREAMFUNCTION, ROW NUMBER ", KL ) ;
             IF PRNT EQL 3 THEN SIMWRITE(LINE,
                         "INITIAL STREAMFUNCTION, ROW NUMBER ",KL);
             SIMWRITE (LINE, VAR[KL] ) ;
             END ;
        MODE := SAVEMODE ;
        END ;
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    SYSTEMPARAMETERS ;
        BEGIN
        MODE := TRUE ;
        NREY := 40.0 ;
        DT := 0.04   ;
        TLMT := 0.08;
        NETA := 9 ;
                        % NUMBER OF THETA MESH POINTS
        NTSI := 17 ;
                        % NUMBER OF RADIAL MESH POINTS
        ETAMAX := 1.0 ;
        TSIMAX := 1.5 ;
        PINC := 1 ;
        END ;
%   ***********************************************************************
%
                        PE REAL SUBROUTINE
    COMPUTESYSTEMCONSTANTS ;
        BEGIN
        MODE := TRUE ;
        PI := 4 * ARCTAN(1) ;
        BCS1 := BOOLEAN(8000000000000000(16) )  ;
        BC11 := BOOLEAN(0C000000000000000(16) )  ;
        AD :=  BOOLEAN(0000000000000001(16)) ;
        DETA := ETAMAX /(NETA - 1) ;
        DTSI := TSIMAX /(NTSI - 1) ;
        HX := 1.0 / (DTSI * DTSI) ;
        HY := 1.0 / (DETA * DETA) ;
        LL1 := NTSI - 2;
                        % LOOP LIMIT ONE, THE NUMBER OF * INTERNAL *
                        %    MESH POINTS IN THE RADIAL  (TSI)  DIRECTION
        LL2 := NTSI - 3;
                        % LOOP LIMIT TWO, INDICATES TO THE PROGRAM
```

```
                %    WHEN THE LAST SECTION HAS BEEN PROCESSED,  00012200
                %    OR,  WHEN THE LAST ROW HAS BEEN PROCESSED  00012300
                %    IN THE THOMAS METHOD.                      00012400
IF LL1 LEQ 62 THEN LL3 := LL1                                   00012500
                ELSE BEGIN                                      00012600
                     IF LL1 LEQ 124 THEN LL3 := LL1/2           00012700
                ELSE BEGIN                                      00012800
                     IF LL1 LEQ 186 THEN LL3 := LL1/3           00012900
                ELSE BEGIN                                      00013000
                     IF LL1 LEQ 248 THEN LL3 := LL1/4           00013100
                ELSE BEGIN  GO TO FINI END                      00013200
                     END  END  END ;                            00013300
                % LOOP LIMIT THREE, THE NUMBER OF INTERNAL      00013400
                %    RADIAL MESH POINTS IN A SECTION, GIVEN     00013500
                %    HERE AT ITS MAXIMUM VALUE OF 62.  THIS     00013600
                %    NUMBER MUST BE DETERMINED BY THE USER TO   00013700
                %    FIT THE NET LENGTH SO THAT  LL1/LL3  IS    00013800
                %    A WHOLE NUMBER.                            00013900
LL4 := NETA - 2 ;                                               00014000
                % LOOP LIMIT FOUR. INDICATES THE NUMBER OF      00014100
                %    INTERNAL MESH POINTS IN THE CIRCUMFERENTIAL00014200
                %    (ETA) DIRECTION. THE MAXIMUM NUMBER IS 62  00014300
LL5 := NETA - 3 ;                                               00014400
                % LOOP LIMIT FIVE, INDICATES WHEN THE LAST      00014500
                %    COLUMN HAS BEEN PROCESSED IN THE           00014600
                %    THOMAS METHOD                              00014700
LL6 := NTSI - 1 ;                                               00014800
KE := PI * DTSI ;                                               00014900
DX := EXP(KE) ;                                                 00015000
IDX := 1.0/DX ;                                                 00015100
                %                                               00015200
                % CALCULATE THE BOOLEAN MASKS NEEDED FOR THE    00015300
                %    ARRAY WIDTH USED.                          00015400
                %                                               00015500
     BEGIN                                                      00015600
     CINT J ;                                                   00015700
     BC := FALSE ;                                              00015800
     LOOP J:=1,1,LL4   DO                                       00015900
          BEGIN                                                 00016000
          BC := BC OR BCS1 ;                                    00016100
          BC := REVR(1,BC) ;                                    00016200
          END ;                                                 00016300
     BCS := REVR(1,BC) ;                                        00016400
     BCS := BCS OR BC11 ;                                       00016500
     END ;                                                      00016600
                % THE BOOLEAN VARIABLE BC MUST REFLECT THE      00016700
                %    WIDTH OF THE NET.  A "1" SHOULD BE PLACED  00016800
                %    IN EACH BIT POSITION WHICH CORRESPONDS     00016900
                %    TO A PE CONTAINING ONE OF THE INTERNAL MESH00017000
                %    POINTS. ALL OTHER BITS MUST BE ZERO.       00017100
     BEGIN                                                      00017200
     CINT J;                                                    00017300
     BCT := FALSE ;                                             00017400
     LOOP J := 1,1,LL3 DO                                       00017500
         BEGIN                                                  00017600
         BCT := BCT OR BCS1 ;                                   00017700
         BCT := REVR(1,BCT);                                    00017800
         END;                                                   00017900
     END;                                                       00018000
TERM4I := 0.25 /(DTSI * DETA) ;                                 00018100
```

```
      TERMB  := 2.0  /(NREY * DETA * DETA) ;                          00018200
      TERMB2 :=  2.0 * TERMB ;                                        00018300
      TERMD  := 2.0  /(NREY * DTSI * DTSI) ;                          00018400
      TERMD2 :=  2.0 * TERMD ;                                        00018500
      DTI2   :=  2.0/DT ;                                             00018600
%                                                                     00018700
      MODE := TRUE ;
      MODE := (PEN LEQ (NETA - 1) / 2 ) ;
      ETA := PEN ;
      MODE := TRUE ;
      MODE := (PEN GTR (NETA - 1) / 2 ) ;
      ETA := (NETA - 1 ) - PEN ;
      MODE := BC ;                                                    00018800
      M := ETA * PI * DETA ;
      S := SIN(M) ;                                                   00019100
      MODE := TRUE ;                                                  00019200
      END ;                                                           00019300
%  ***************************************************************** 00019400
%                                                                     00019500
                    PE REAL SUBROUTINE                                00019600
COMPUTEADICOEFFICIENTSANDSKEW (CREAL E, CNPOINT TERME,                00019700
                    CNPOINT COFA, PCPOINT KFA, PCPOINT KFB,           00019800
                    PCPOINT KFC, PCPOINT KFD ) ;                      00019900
      BEGIN         % SKEW COEFFICIENTS                               00020000
      CINT K,J,L ;                                                    00020100
      PREAL TERMA, TERMC ;                                            00020200
      BOOLEAN BCI ;                                                   00020300
      MODE := BC ;                                                    00020400
                    % IF THE NET LENGTH (RADIAL DIRECTION) IS         00020500
                    %    LONGER THAN 62, THE NET MUST BE SEPARATED    00020600
                    %    INTO SEVERAL EQUAL LENGTH SECTIONS. A        00020700
                    %    SECTION MAY AT MOST BE 62 ROWS LONG          00020800
                    %    (LL3=62) AND 62 WIDE (LL4=62) .  THE         00020900
                    %    PROGRAM TREATS ONE SECTION AT A TIME UNTIL   00021000
                    %    ALL SECTIONS HAVE BEEN SOLVED FOR THE        00021100
                    %    NEW VORTICITY VALUES.  LL2 TELLS THE         00021200
                    %    PROGRAM WHEN THE LAST SECTION HAS BEEN       00021300
                    %    PROCESSED.                                   00021400
      LOOP  K := 0,LL3, LL2   DO                                      00021500
           BEGIN                                                      00021600
           BCI := BC ;                                                00021700
           MODE := BCI ;                                              00021800
           LOOP  J := 1, 1, LL3   DO                                  00021900
                    % THERE ARE LL3 ROUTES IN EACH                    00022000
                              %    SECTION TO SKEW THE ARRAY.         00022100
                BEGIN                                                 00022200
                L := K + J  ;                                         00022300
                TERMA :=-(PSI[L+1] -PSI[L-1]) * TERM4I ;              00022400
                TERMC :=-(RTL(1,TRUE,PSI[L])-RTR(1,TRUE,PSI[L]))      00022500
                     * TERM4I ;                                       00022600
                TERME[L] := E*E*DTI2 ;                                00022700
                COFA[L] := TERME[L] + TERMB2 ;                        00022800
                     % " A "                                          00022900
```

```
                E := E * DX ;                                                   00023000
                KFA[L]  := RTR(J-1,    ,TERMA - TERMB  );                       00023100
                     % " C "                                                    00023200
                KFB[L] := RTR(J-1,     ,-TERMA - TERMB ) ;                      00023300
                     % " B "                                                    00023400
                KFC[L]:= RTR(J-1,     ,(TERMC+TERMD) * ZETA[L-1]                00023500
                     +(TERME[L]-TERMD2) * ZETA[L]                               00023600
                     +(-TERMC+TERMD) * ZETA[L+1]);                              00023700
                     % " D "                                                    00023800
                KFD[L] := COFA[L] ;                                             00023900
                BCI := REVR(1,BCI) ;                                            00024000
                MODE := BCI ;                                                   00024100
                END ;                                                           00024200
           END ;                                                                00024300
      MODE := TRUE ;                                                            00024400
      END ;                                                                     00024500
%  ***************************************************************************00024600
%                                                                               00024700
                     PE REAL SUBROUTINE                                         00024800
   THOMASONROWS (PCPOINT KFA, PCPOINT KFB, PCPOINT KFC, PCPOINT KFD ) ;         00024900
      BEGIN          % APPLY THE THOMAS METHOD TO EACH ROW IN THE              00025000
                     %    SECTION SIMULTANEOUSLY THUS INVERTING                00025100
                     %    LL3 MATICIES AT A TIME.                              00025200
                     %                                                         00025300
                     % REFERENCE TEXT: LAPIDUS, L. , DIGITAL                  00025400
                     %    COMPUTATION FOR CHEMICAL ENGINEERS,                  00025500
                     %    MC GRAW-HILL, 1962, P.254.                           00025600
      CINT J,K ;                                                                00025700
      MODE := TRUE ;                                                            00025800
      LOOP  K := 0,LL3, LL2   DO                                                00025900
           BEGIN                                                                00026000
           PINT  R ;                                                            00026100
           BOOLEAN BCI ;                                                        00026200
                BEGIN                % WORKING FORWARD COMPUTE                00026300
                                     %    INTERMEDIATE VALUES                 00026400
                CINT  N ;                                                       00026500
                                     % BC CARRIES THE BIT PATTERN             00026600
                                     %    REFLECTING THE WIDTH OF THE         00026700
                                     %    NET. THIS INFORMATION IS            00026800
                                     %    PASSED TO BCI FOR USE IN            00026900
                                     %    ENABLING THE PROPER                 00027000
                                     %    PROCESSING ELEMENTS WHILE           00027100
                                     %    OPERATING ON THE SKEWED             00027200
                                     %    ARRAY.                              00027300
                BCI := BCT;                                                     00027400
                MODE := TRUE ;                                                  00027500
                R :=  PEN + K ;                                                 00027600
                                     % R IS LIKE A POINTER WHICH              00027700
                                     %    POINTS TO THE PROPER                00027800
                                     %    VECTOR ELEMENT IN EACH PE           00027900
                MODE := BCI ;                                                   00028000
                                     % W(1) = A(1)                            00028100
                KFB[R] := KFB[R] / KFD[R] ;                                     00028200
                                     % Q(1) = B(1) / W(1)                      00028300
                KFC[R] := KFC[R] / KFD[R] ;                                     00028400
                                     % G(1) = D(1) / W(1)                      00028500
                LOOP  J := 2,1,LL4 DO                                           00028600
                     BEGIN                                                      00028700
                     MODE := TRUE ;                                             00028800
                                     % SHIFT INDEXING AND ENABLING             00028900
```

```
                                        %    PARAMETERS
                        R  := RTR(1,MODE,R) ;
                        BCI:= REVR(1,BCI)   ;
                        MODE := BCI ;
                        KFD[R]:=  KFD[R]  - KFA[R]
                                        *RTR(1,  ,KFB[R+1]) ;
                                        % W(R) = A(R) - C(R) * Q(R-1)
                        KFB[R] := KFB[R] / KFD[R] ;
                                        % Q(R) = B(R) / W(R)
                        KFC[R]:=(KFC[R]- KFA[R]
                                        * RTR(1,  ,KFC[R+1])   )
                                        / KFD[R] ;
                                        % G(R)=( D(R)-C(R)*G(R-1) )/W(R)
                        END ;
                    END ;
                    BEGIN               % WORK BACKWARD TO GET SOLUTION
                                        %    VECTORS
                    KFD[R] := KFC[R] ;
                                        % X(N) = G(N)
                    LOOP   J := 1,1,LL5    DO
                        BEGIN
                        MODE := TRUE ;
                        R := RTL(1,MODE,R) ;
                        BCI := REVL(1,BCI) ;
                        MODE := BCI ;
                        KFD[R] := KFC[R]-KFB[R]
                                        *RTL(1,  ,KFD[R-1]) ;
                                        % X(R) = G(R) - Q(R) * X(R+1)
                                        % NEW VORTICITY VALUES STORED
                                        %    IN SKEWED FORM IN KFD[R]
                        END ;
                    END ;
                END ;
            MODE := TRUE ;
            END ;
% ***********************************************************************
%
                        PE REAL SUBROUTINE
UNSKEWANDSTOREINTERMEDIATEVORTICITY (PCPOINT KFD) ;
            BEGIN           % UNSKEW RESULTS AND OVERLAY VORTICITY NET
                            %     WITH THE NEW VALUES
            CINT J,K,L ;
            MODE := BC ;
            LOOP  K := 0, LL3,LL2   DO
                BEGIN
                LOOP  J := 1,1,LL3  DO
                    BEGIN
                    L := K + J ;
                    ZETA[L] := RTL(J-1,TRUE,KFD[L]) ;
                    END ;
                END ;
            MODE := TRUE ;
            END ;
                        %  W[R] ::= KFD[R]      OR  D2
                        %  Q[R] ::= KFB[R]  OR  B
                        %  G[R] ::= KFC[R]  OR  D1
                        %  X[R] ::= KFD[R]  OR  D2,W[R]
% ***********************************************************************
%
                        PE REAL SUBROUTINE
```

```
COMPUTENEWADICOEFFICIENTS (CNPOINT TERME, CNPOINT COFA,                  00035000
                    PCPOINT KFA, PCPOINT KFB, PCPOINT KFC,               00035100
                    PCPOINT KFD ) ;                                      00035200
     BEGIN          % DEVELOP NEW COEFFICIENTS                           00035300
     CINT J,K,L ;                                                        00035400
     PREAL TERMA, TERMC ;                                                00035500
     MODE := BC ;                                                        00035600
     LOOP  J := 1, 1, LL1  DO                                            00035700
          BEGIN                                                          00035800
          TERMA :=-(PSI[J+1] -PSI[J-1]) * TERM4I ;                       00035900
          TERMC :=-(RTL(1,TRUE,PSI[J]) - RTR(1,TRUE,PSI[J]))            00036000
                    * TERM4I ;                                           00036100
          COFA[J] := TERME[J] + TERMD2 ;                                 00036200
                    % "A"                                                00036300
          KFA[J]  := (-TERMA+TERMB)* RTR(1,TRUE,ZETA[J])                 00036400
                    +(TERME[J] - TERMB2 ) * ZETA[J]                      00036500
                    +(TERMA+TERMB) * RTL(1,TRUE,ZETA[J])  ;              00036600
                    % "D"                                                00036700
          KFB[J]:= (TERMC-TERMD) ;                                       00036800
                    % "B"                                                00036900
          KFC[J]:= (-TERMC-TERMD) ;                                      00037000
                    % "C"                                                00037100
          END ;                                                          00037200
     MODE := TRUE ;                                                      00037300
     END ;                                                               00037400
%  ***************************************************************************00037500
%                                                                        00037600
                    PE REAL SUBROUTINE                                   00037700
THOMASONCOLUMNSANDSTOREUPDATEDVORTICITY (CNPOINT TERME,                  00037800
                    CNPOINT COFA, PCPOINT KFA, PCPOINT KFB,              00037900
                    PCPOINT KFC, PCPOINT KFD ) ;                         00038000
     BEGIN                                                               00038100
     CINT L,J ;                                                          00038200
          BEGIN     % THOMAS METHOD ON EACH COLUMN                       00038300
          MODE := BC ;                                                   00038400
          KFD[1]   := COFA[1] ;                                          00038500
                    % W(1) = A(1), (747+1)                               00038600
          KFB[1]   := KFB[1]   / KFD[1] ;                                00038700
                    % Q(1) = B(1) / W(1), (249+1), (747+1)               00038800
          KFA[1] := ( KFA[1] - KFC[1] * ZETA[0] ) / KFD[1] ;             00038900
                    % G(1) = D(1) / W(1), (747+1)                        00039000
          LOOP  J := 2, 1, LL1  DO                                       00039100
               BEGIN                                                     00039200
               KFD[J]:=COFA[J]-KFC[J]*KFB[J-1] ;                         00039300
                    % W(R) = A(R) - C(R) * Q(R-1) , (249-1)              00039400
               KFB[J]:=KFB[J] / KFD[J] ;                                 00039500
                    % Q(R) = B(R) / W(R)                                 00039600
               KFA[J]:=(KFA[J]-KFC[J]*KFA[J-1])/KFD[J] ;                 00039700
                    % G(R) = (D(R) - C(R) * G(R-1) ) / W(R)              00039800
               END ;                                                     00039900
          END ;                                                          00040000
          BEGIN     % ENTER RESULTS IN VORTICITY NET                    00040100
          ZETA[LL1] := KFA[LL1] ;                                        00040200
                    % X(N) = G(N)                                        00040300
          LOOP  J := 1, 1, LL2  DO                                       00040400
               BEGIN                                                     00040500
               L := LL1 - J ;                                            00040600
               ZETA[L] := KFA[L] - KFB[L] * ZETA[L+1] ;                  00040700
                    % X(R) = G(R) - Q(R) * X(R+1),  (747+1)              00040800
               END ;                                                     00040900
```

```
            MODE := TRUE ;                                                  00041000
$*          DISPLAY 0,1;                                                    00041100
            END ;                                                           00041200
      END ;                                                                 00041300
%  ***************************************************************************00041400
%                                                                           00041500
                        PE REAL SUBROUTINE                                  00041600
   VORTICITY ( PCPOINT PSI, PCPOINT ZETA ) ;                                00041700
      BEGIN             % COMPUTE NEW VALUES OF THE VORTICITY USING        00041800
                        %    THE PEACEMAN RACHFORD ALTERNATING DIRECTION   00041900
                        %    IMPLICIT METHOD  ( ADI )                      00042000
      CREAL E ;                                                             00042100
      PREAL VECTOR KFA[ 20],KFB[ 20],KFC[ 20],KFD[ 20];                     00042200
      CREAL VECTOR TERME[ 20] , COFA[ 20] ;                                 00042300
                        % PARAMETERS REQUIRED, DTSI, DETA, NREY, PI, DX,   00042400
                        %    DT, BC, LL1,LL2,LL3,LL4,LL5,BCI,              00042500
                        %    TERM4I,TERMB,TERMB2,TERMD,TERMD2,             00042600
                        %    DTI2,BCT                                      00042700
                                                                            00042800
                        %                                                   00042900
      MODE := BC ;                                                          00043000
      E := PI * DX ;                                                        00043100
$*    DISPLAY 0,1;                                                          00043200
                        % IN THE FIRST HALF TIME STEP THE VORTICITY        00043300
                        %    IS IMPLICIT ACROSS THE ROWS. BY SKEWING       00043400
                        %    THE ARRAYS OF COEFFICIENTS, ALL ROWS CAN      00043500
                        %    BE OPERATED ON SIMULTANEOUSLY. THE            00043600
                        %    THOMAS METHOD IS APPLIED SEQUENTIALLY TO      00043700
                        %    EACH ELEMENT IN A ROW.                        00043800
                        %                                                  00043900
      MODE := TRUE ;                                                        00044000
      COMPUTEADICOEFFICIENTSANDSKEW (E,TERME,COFA,KFA,KFB,KFC,KFD) ;        00044100
      THOMASONROWS (KFA,KFB,KFC,KFD) ;                                      00044200
      UNSKEWANDSTOREINTERMEDIATEVORTICITY (KFD) ;                           00044300
                        %                                                   00044400
                        % IN THE SECOND HALF TIME STEP THE VORTICITY       00044500
                        %    IS IMPLICIT WITHIN THE COLUMNS.  WITH         00044600
                        %    THE COEFFICIENTS STORED STRAIGHT, ALL         00044700
                        %    COLUMNS CAN BE OPERATED ON SIMULTANEOUSLY.    00044800
                        %    THE THOMAS METHOD IS APPLIED SEQUENTIALLY     00044900
                        %    TO EACH ELEMENT IN A COLUMN.                  00045000
                        %                                                  00045100
      COMPUTENEWADICOEFFICIENTS (TERME,COFA,KFA,KFB,KFC,KFD) ;              00045200
      THOMASONCOLUMNSANDSTOREUPDATEDVORTICITY (TERME,COFA,                  00045300
```

```
                    KFA,KFB,KFC,KFD) ;                                    00045400
     MODE := TRUE ;                                                       00045500
     END ;                                                                00045600
%  ***************************************************************************00045700
%                                                                         00045800
                    PE INTEGER SUBROUTINE                                 00045900
TURN AS RGA(CINT P, PINT I);                                              00046000
     BEGIN                                                                00046100
     RGA  := I ; CAR1  := P ;                                             00046200
     CODE BEGIN                                                           00046300
          LIT(2)      =1,0,1 ;                                            00046400
          SHABR       0(1) ;                                              00046500
          CSHL(1)     24;                                                 00046600
          CLRA ;                                                          00046700
          COR(1)     $C2 ;                                                00046800
          SHABL       1 ;                                                 00046900
          RTAR        2 ;                                                 00047000
          TXEFAM(1)   ,-3;                                                00047100
          CSHR(1)    24 ;                                                 00047200
          RTAL        1(1);                                               00047300
          END CODE ;                                                      00047400
     END ; % OF TURN                                                      00047500
%  ***************************************************************************00047600
%                                                                         00047700
                    SUBROUTINE                                            00047800
SETPT1 (PCPOINT A, PCPOINT S, PCPOINT IS,CINT OUT PP,                     00047900
                    CINT NX,CREAL HX,CREAL HY) ;                          00048000
     BEGIN                                                                00048100
          %SETPT1 CALCULATES FOUR PARAMETERS REQUIRED BY                  00048200
          %POISSONDIRECT, A,S,IS,INDEX.  THESE PARAMETERS                 00048300
          %ARE VECTORS WHICH REQUIRE CN/128 LINES OF PEM                  00048400
          %EACH.  CN IS THE NUMBER OF ROWS IN THE MESH, NOT               00048500
          %PEM ROWS. INDEX IS A PINT VECTOR AND IS THUS A                 00048600
          %GLOBAL VARIABLE FROM THE MAIN PROGRAM.  NX EQUALS              00048700
          %CN-1.  HX AND HY ARE THE HORIZONTAL AND VERTICAL               00048800
          %COEFFICIENTS RESPECTIVELY.  PP IS THE LOG BASE 2 OF NX.        00048900
          %SETPT1 MUST BE CALLED EVERY TIME THE MESH SIZE CHANGES         00049000
          %OR HX AND(OR) HY CHANGE.                                       00049100
     CINT I,R,I1,I2,KN ;                                                  00049200
     PINT L,Q,LX ;
     PREAL QT, CO ;
     CREAL H,X ;                                                          00049500
     QT := NTSI-1 ;                                                       00049600
     PP := LN(QT) / LN(2.0) ;                                             00049700
     L  := PEN ;                                                          00049800
     X := PI / NX ;                                                       00049900
     H  := HX / HY ;                                                      00050000
     KN := NX.[16:4] ;                                                    00050100
     LOOP I  := 0,1,KN DO                                                 00050200
          BEGIN                                                           00050300
          MODE := TRUE ;
          MODE := (L LEQ LL6 / 2 ) ;
          LX := L ;
          CO :=  COS(LX * X) ;
          MODE := TRUE ;
          MODE := (L GTR LL6 / 2 ) ;
          LX := LL6 - L ;
          CO := -COS(LX * X) ;
          MODE := TRUE ;
          S[I] := SIN(LX*X) ;
```

```
          IS[I]  := S[I] +S[I] ;                                          00050500
          IF IS[I] NEQ 0 THEN BEGIN IS[I]:= -1.0 / IS[I] ; END;           00050600
          A[I] := -2.0 * (H-H * CO + 1 ) ;
          INDEX[I]  := TURN(PP-1,L);                                      00050800
          L  := L + 64 ;                                                  00050900
          END ;                                                           00051000
     END ;                                                                00051100
%  ***************************************************************************00051200
%                                                                         00051300
                    SUBROUTINE                                            00051400
FOURIER(PCPOINT P,CINT K,CINT CN,CINT LN,BOOLEAN FMODE,                   00051500
                    PCPOINT DS,PCPOINT IS) ;                              00051600
     BEGIN                                                                00051700
     CINT J1 ;                                                            00051800
     BOOLEAN SMODE ;                                                      00051900
     CINT I,L,J,IL,N2,ILJ,CJ,KM,N3,IT,IP,N11,I2,I1,IM,IQ,KP,KM1,GL;       00052000
     PINT PI;                                                             00052100
     PREAL ODD1,ODD2,ODD3;                                                00052200
     CREAL RR,RR1,C,S   ;                                                 00052300
     J  :=(CN-1)*K;                                                       00052400
     L:=J.[16:47];  PPP:=1;                                               00052500
     PI  :=0;                                                             00052600
     IF  ( PEN EQL OFF+1 ) THEN PI := -1 ;                                00052700
     MODE  := FMODE ;                                                     00052800
     KM1  := K-1 ;                                                        00052900
     KP  := K + K ;                                                       00053000
     N3  := CN - 1 ;                                                      00053100
     N2 := N3.[16:47];                                                    00053200
     LOOP ILJ  := 0,1,KM1 DO                                              00053300
          BEGIN                                                           00053400
          KM  := K + ILJ ;                                                00053500
          LOOP I  := K,K,L-K DO                                           00053600
               BEGIN                                                      00053700
               IL  := I - ILJ ;                                           00053800
               ODD1  := P[I+ILJ+PI];                                      00053900
               P[PI+I+ILJ]  := ODD1 + P[PI+J-IL] ;                        00054000
               P[PI+J-IL]  := ODD1 - P[PI+J-IL] ;                         00054100
               END;                                                       00054200
          P[PI+L+ILJ]  := P[PI+L+ILJ] + P[PI+L+ILJ] ;                     00054300
          IL  := K - ILJ ;                                                00054400
          P[PI+ILJ]  := - P[PI+J-IL] - P[PI+J-IL] ;                       00054500
          ODD3 := P[PI+KM];                                               00054600
          P[PI+KM] := -ODD3 - ODD3 ;                                      00054700
          IL := 0 ;                                                       00054800
          LOOP I := KP,KP,L-KP DO                                         00054900
               BEGIN                                                      00055000
               IL := IL + 2 ;                                             00055100
               CJ := I + ILJ ;                                            00055200
               ODD1 := P[PI+I+KM] - ODD3 ;                                00055300
               GL := IL .[16:42] ;                                        00055400
               SMODE := MODE ; MODE := TRUE ;                             00055500
               RR := GRABONE(DS[GL],IL) ;                                 00055600
               GL := (N2-IL).[16:42];                                     00055700
               RR1 := GRABONE(DS[GL],N2-IL);%OME UP IF N2]64              00055800
               MODE := SMODE ;                                            00055900
               ODD2 := RR * P[PI+CJ] - RR1 * ODD1 ;                       00056000
               ODD1 := RR1 * P[PI+CJ] + RR * ODD1 ;                       00056100
               P[PI+CJ] := P[PI+J+KM-I] - P[PI+J-I-K+ILJ] ;               00056200
               ODD3 := P[PI+I+KM] ;                                       00056300
               P[PI+I+KM] := ODD2 - P[PI+J-I+ILJ];                        00056400
```

```
            P[PI+J-I+KM] := ODD2 + P[PI+J-I+ILJ];                    00056500
            P[PI+J-I+ILJ] := P[PI+CJ] - ODD1 ;                       00056600
            P[PI+CJ] := P[PI+CJ] + ODD1 ;                            00056700
            END ;                                                    00056800
      ODD1 := P[PI+L+KM] ;                                           00056900
      P[PI+L+KM] := P[PI+L+ILJ] ;                                    00057000
      P[PI+L+ILJ] := ODD1 + ODD1 ;                                   00057100
      LOOP I := ILJ,K,L-K+ILJ DO                                     00057200
            BEGIN                                                    00057300
            ODD1 := P[PI+I] ;                                        00057400
            P[PI+I] := ODD1 + P[PI+L+I] ;                            00057500
            P[PI+L+I] := ODD1 - P[PI+L+I] ;                          00057600
            END;                                                     00057700
      MODE := TRUE ;                                                 00057800
      END;                                                           00057900
                  % SCRAMBLING                                       00058000
IL := 0 ;                                                            00058100
KM :=K+KM1      ;                                                    00058200
LOOP I := KP,KP,J-KP DO                                              00058300
      BEGIN                                                          00058400
      IL := IL + 1 ;                                                 00058500
      GL := IL .[16:42] ;                                            00058600
      ODD3 := INDEX[GL] ;                                            00058700
      CJ :=  GRABONE(ODD3      ,IL) ;                                00058800
      IF IL LSS CJ THEN                                              00058900
            BEGIN                                                    00059000
            CJ := CJ * KP ;                                          00059100
            LOOP ILJ := 0,1,KM DO                                    00059200
                  BEGIN                                              00059300
                  ODD1 := P[    I+ILJ] ;                             00059400
                  P[    I+ILJ] := P [  CJ+ILJ] ;                     00059500
                  P[    CJ+ILJ] := ODD1    ;                         00059600
                  END ;                                              00059700
            END ;                                                    00059800
      END ;                                                          00059900
IT := KP + KP ;                                                      00060000
IP := N3 ;                                                           00060100
LOOP I := 2,1,LN  -1 DO                                              00060200
      BEGIN                                                          00060300
      IP:=IP.[16:47] ;                                               00060400
      N11 := 0 ;                                                     00060500
      I2 := 1 ;                                                      00060600
      CJ := IT.[16:47] ;                                             00060700
      IT := IT + IT ;                                                00060800
      LOOP IL := 0,CJ,CJ DO                                          00060900
            BEGIN                                                    00061000
            LOOP KM := IL,KP,IL+CJ-KP DO                             00061100
                  BEGIN                                              00061200
                  GL := (N2-N11) .[16:42] ;                          00061300
                  C := GRABONE(DS[GL],N2-N11) * I2 ;                 00061400
                  GL := N11 .[16:42] ;                               00061500
                  S := GRABONE(DS[GL],N11) ;                         00061600
                  LOOP J1 := 0,IT,J-IT DO                            00061700
                        BEGIN                                        00061800
                        MODE := FMODE ;                              00061900
                        LOOP ILJ := 0,1,KM1 DO                       00062000
                              BEGIN                                  00062100
                              IM := J1 + KM + ILJ ;                  00062200
                              IQ := IM + CJ + CJ ;                   00062300
                              ODD1 := P[PI+IQ]*C-P[PI+IQ+K] * S ;    00062400
```

```
                              ODD2 := P[PI+IQ+K]*C+P[PI+IQ] * S ;    00062500
                              P[PI+IQ] := P[PI+IM] - ODD1 ;          00062600
                              P[PI+IQ+K] := P[PI+IM+K] - ODD2 ;      00062700
                              P[PI+IM] := P[PI+IM] + ODD1 ;          00062800
                              P[PI+IM+K] := P[PI+IM+K] + ODD2 ;      00062900
                              MODE := TRUE ;                         00063000
                              END ;                                  00063100
                         END ;                                       00063200
                    N11 := N11 + IP ;                                00063300
                    END ;                                            00063400
               I2 := -1 ;                                            00063500
               IP := - IP ;                                          00063600
               END ;                                                 00063700
          END ;                                                      00063800
     I := 0 ;                                                        00063900
     LOOP I1 := 1,1,N2 DO                                            00064000
          BEGIN                                                      00064100
          I := I + K ;                                               00064200
          GL := I1 .[16:42] ;                                        00064300
          RR := GRABONE(IS[GL],I1);                                  00064400
          MODE := FMODE ;                                            00064500
          LOOP ILJ := 0,1,KM1 DO                                     00064600
               BEGIN                                                 00064700
               ODD1 := P[PI+I+ILJ] - P[PI+J-I+ILJ] ;                 00064800
               ODD2 := ( P[PI+I+ILJ] + P[PI+J-I+ILJ] ) * RR ;        00064900
               P[PI+I+ILJ] := ODD2 + ODD1 ;                          00065000
               P[PI+J-I+ILJ] := ODD2 - ODD1 ;                        00065100
               MODE := TRUE ;                                        00065200
                                                                     00065300
               END ;                                                 00065400
          END ;                                                      00065500
     END ;                                                           00065600
% ***************************************************************** 00065700
%                                                                    00065800
                         SUBROUTINE                                  00065900
   POISSONDIRECT (PCPOINT P,PCPOINT C,PCPOINT A,PCPOINT S,CINT CN,   00066000
                         CINT CM,CREAL HX,CREAL HY,PCPOINT IS,       00066100
                         CINT LN ) ;                                 00066200
%%%%%%%%%%%%%%%%  FUNCTION  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  00066300
%                                                                    00066400
%     THIS SUBROUTINE SOLVES POISSON' EQUATION USING A DIRECT        00066500
%     METHOD SIMILAR TO HOCKNEY'S PROGRAM [1].                       00066600
%     C[I,J] = HX*(P[I+1,J]+P[I-1,J]-2*P[I,J] +                      00066700
%              HY*(P[I,J+1]+P[I,J-1]-2*P[I,J]).                      00066800
%     THE CHARGE, C, AND THE BOUNDARY POINTS OF THE POTENTIAL,       00066900
%     P, ARE USED TO CALCULATE THE INTERIOR POINTS OF POTENTIAL.     00067000
%                                                                    00067100
%%%%%%%%%%%%%%%%  PARAMETERS  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  00067200
%                                                                    00067300
%     P IS THE POTENTIAL.  C IS THE CHARGE.                          00067400
%     A CONTAINS THE DIAGONAL ELEMENTS OF THE TRIDIAGONAL            00067500
%     MATRICES SOLVED IN THE SECTION OF THIS ROUTINE CALLED          00067600
%     CRED.  S CONTAINS THE VALUES OF THE SIN WHICH ARE NEEDED       00067700
%     BY THE FOURIER TRANSFORM PART OF THE ALGORITHM.  IS IS         00067800
%     ALSO NEEDED BY FOURIER.                                        00067900
%     INDEX IS GLOBAL FROM THE MAIN PROGRAM AND IS NEEDED BY         00068000
%     FOURIER.  A, S, IS, AND INDEX ARE CALCULATED BY SETPT1.        00068100
%     P HAS CN ROWS AND CM COLUMNS.  HX(HY) IS THE HORIZONTAL(       00068200
%     VERTICAL) COEFFICENT OF POISSON'S EQUATION.  LN EQUALS         00068300
%     THE LOG BASE 2 OF CN-1.                                        00068400
```

```
%                                                                       00068500
%%%%%%%%%%%%%%%%  METHOD  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  00068600
%                                                                       00068700
%     A FOURIER ANALYSIS IS DONE TO EACH COLUMN BY SUBROUTINE           00068800
%     FOURIER.  THIS IS FOLLOWED BY SOLVING THE TRIDIAGONAL             00068900
%     MATRICES, ONE PER ROW.  FINALLY A FOURIER SYNTHESIS IS            00069000
%     PERFORMED ON EACH ROW TO GIVE ANSWER.  FOR FURTHER DETIAL          00069100
%     SEE [2].                                                          00069200
%                                                                       00069300
%%%%%%%%%%%%%%%%%%%%  REFERENCES  %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  00069400
        BEGIN                                                           00069500
        CINT K,CM2,CN2,CMIK,CNIK,CI,CJ,CL,IL,L1,L2,L3,L4,KM1,KP,KC;     00069600
        PINT PI,PJ;                                                     00069700
        CREAL POTFAC; BOOLEAN FMODE ;                                   00069800
        PCPOINT STORE ;                                                 00069900
        PREAL PR, E, Q, T ;                                             00070000
        MODE := TRUE ;                                                  00070100
        CMIK := CM .[16:42] ;                                           00070200
        CI := 0 ;                                                       00070300
        IF CM.[58:6] GTR 0 THEN CI := 1 ;                               00070400
        CN2 := CN - 2 ; POTFAC := 0.125 / ((CN-1)*HY) ;                 00070500
        CM2 := CM - 2 ;                                                 00070600
        K := CMIK + CI ;                                                00070700
        KM1 := K - 1 ;                                                  00070800
        KP := K + K ;                                                   00070900
        STORE := GETPEB(KP+K) ;                                         00071000
        KC := K * CN2 ; PI := 0 ;                                       00071100
        CI := CM.[58:6] ;                                               00071200
        OFF := CI - 2 ;                                                 00071300
        IF CI = 0 THEN OFF := 62 ;                                      00071400
        MODE :=  ( PEN GTR OFF ) ;                                      00071500
        PI := -1 ;                                                      00071600
        MODE := TRUE ;                                                  00071700
        CJ := KC + 1 ;                                                  00071800
        LOOP CI := 0,1,KM1 DO                                           00071900
             BEGIN                                                      00072000
             STORE[CI] := C[CI+K] ;                                     00072100
             STORE[CI+K] := C[CI+KC] ;                                  00072200
             STORE[CI+KP] := P[CI] ;                                    00072300
```

```
     END ;                                                          00072400
LOOP CI  := 1,1,KM1 DO                                              00072500
     BEGIN                                                          00072600
     C[PI+CJ] := C[PI+CJ] - P[CJ+PI+K] * HX ;                       00072700
     CJ := CJ +1 ;                                                  00072800
     C[PI+K+CI] := C[PI+K+CI] - P[PI+CI] * HX ;                     00072900
     END;                                                           00073000
IF PEN GTR 0 AND PEN LSS OFF+1 THEN                                 00073100
     BEGIN                                                          00073200
     C[K] := C[K] -P[0] * HX ;                                      00073300
     C[KC] := C[KC] - P[KC+K] * HX ;                                00073400
     FMODE := MODE ;                                                00073500
     END ;                                                          00073600
PJ := 0 ;                                                           00073700
PI := 0 ;                                                           00073800
MODE :=  ( PEN EQL OFF  ) ;                                         00073900
PI := KM1 ;                                                         00074000
MODE := TRUE ;                                                      00074100
MODE := ( PEN GTR OFF-1 ) ;                                         00074200
PJ := -1 ;                                                          00074300
MODE := TRUE ;                                                      00074400
IF OFF = 1 THEN BEGIN FMODE :=(PEN GTR 0); END;                     00074500
LOOP CI := K,K,KC DO                                                00074600
     BEGIN                                                          00074700
     PR := 0.0 ;                                                    00074800
     MODE :=  ( PEN EQL OFF ) ;                                     00074900
     PR := RTL(1,,P[CI+KM1]);                                       00075000
     MODE := TRUE ;                                                 00075100
     MODE := PEN EQL 1 ;                                            00075200
     PR := RTR(1,,P[CI]) + PR ;                                     00075300
     MODE := TRUE ;                                                 00075400
     MODE := FMODE ;                                                00075500
     P[CI+PI] :=(C[CI+PI] -PR*HY) * POTFAC ;                        00075600
     MODE := TRUE ;                                                 00075700
     LOOP L1:= 1,1,KM1 DO                                           00075800
          BEGIN                                                     00075900
          P[L1+CI+PJ] :=  C[L1+CI+PJ] * POTFAC ;                    00076000
          END;                                                      00076100
     END ;                                                          00076200
FOURIER (P,  K,CN,LN,FMODE,S,IS) ;                                  00076300
             % SKEW POTENTIAL                                       00076400
CI := 0 ;                                                           00076500
LOOP CJ := KP,K,KC DO                                               00076600
     BEGIN                                                          00076700
     CI := CI + 1 ;                                                 00076800
     LOOP IL := CJ,1,CJ+KM1 DO                                      00076900
          BEGIN                                                     00077000
          P[IL] := RTR(CI,,P[IL]) ;                                 00077100
          END ;                                                     00077200
     END ;                                                          00077300
MODE := (PEN LSS CN-1) ;%   CRED                                    00077400
PI := PEN * K ;                                                     00077500
               % SINCE P[0] TO P[KM1] IS ALREADY                   00077600
               %    GARBAGE WE WONT PROTECT IT                     00077700
CNIK := CN.[16:42] - 1 ;IF CNIK LSS 0 THEN CNIK:=0;                 00077800
CJ := CM.[58:6] ;                                                   00077900
IF CJ=0 THEN CJ := 64 ;                                             00078000
LOOP CI := 0,1,CNIK DO                                              00078100
     BEGIN                                                          00078200
     IL := 4 ;                                                      00078300
```

```
      L4 := 0 ;                                                    00078400
      E := 1.0 / A[CI] ;                                           00078500
      Q := P[PI] * E ;                                             00078600
      CL := 64 ;                                                   00078700
      LOOP L1 := 0,1,KM1 DO                                        00078800
           BEGIN                                                   00078900
           IF L1=KM1 THEN CL :=  CJ ;                              00079000
           LOOP L2 := IL,1,CL DO                                   00079100
                BEGIN                                              00079200
                MODE := REVR(1,MODE) ;                             00079300
                A[CI] := RTR(1,,A[CI]) ;                           00079400
                E := 1.0 / ( A[CI] - RTR(1,,E) ) ;                 00079500
                PI := RTR(1,,PI) ;                                 00079600
                Q := (P[PI+L1+L4] - RTR(1,,Q)) * E ;               00079700
                L4 := 0 ;                                          00079800
                END ;                                              00079900
           L4 := -1 ;                                              00080000
           IL := 1 ;                                               00080100
           END;                                                    00080200
      CL := 64 ;                                                   00080300
      E := P[PI+KM1] - A[CI] * Q ;                                 00080400
      P[PI+KM1] := Q ;                                             00080500
      IL := 4 ;                                                    00080600
      L4 := 0 ;                                                    00080700
      LOOP L1 := 0,1,KM1 DO                                        00080800
           BEGIN                                                   00080900
           IF L1=KM1 THEN CL := CJ ;                               00081000
           L3 := KM1 - L1 ;                                        00081100
           LOOP L2 := IL,1,CL DO                                   00081200
                BEGIN                                              00081300
                MODE := REVL(1,MODE) ;                             00081400
                PI := RTL(1,,PI) ;                                 00081500
                A[CI] := RTL(1,,A[CI]) ;                           00081600
                Q := RTL(1,,Q) ;                                   00081700
                E := RTL(1,,E) ;                                   00081800
                T := P[PI+L3+L4] - Q - A[CI] * E ;                 00081900
                Q := E ;                                           00082000
                E := T ;                                           00082100
                P[PI+L3+L4] := Q ;                                 00082200
                L4 := 0 ;                                          00082300
                END ;                                              00082400
           IL := 1 ; L4 := 1 ;                                     00082500
           END ;                                                   00082600
      PI := PI + 64 * K ;                                          00082700
      END ;                                                        00082800
MODE := TRUE ;%  UNSKEW POTENTIAL                                  00082900
CI := 0 ;                                                          00083000
LOOP CJ := KP,K,KC  DO                                             00083100
      BEGIN                                                        00083200
      CI := CI + 1 ;                                               00083300
      LOOP IL := CJ,1,CJ+KM1 DO                                    00083400
           BEGIN                                                   00083500
           P[IL] := RTL(CI,,P[IL]) ;                               00083600
           END ;                                                   00083700
      END ;                                                        00083800
FOURIER (P,  K,CN,LN,FMODE,S,      IS);                            00083900
LOOP CI := 0,1,KM1 DO                                              00084000
      BEGIN                                                        00084100
      C[CI+K] := STORE[CI] ;                                       00084200
      C[CI+KC] := STORE[CI+K] ;                                    00084300
```

```
          P[CI] := STORE[CI+KP] ;                                        00084400
          END ;                                                          00084500
     END ;                                                               00084600
% ***************************************************************************00084700
%                                                                        00084800
                    PE REAL SUBROUTINE                                   00084900
SURFACEVORTICITY (PCPOINT PSI, PCPOINT ZETA) ;                           00085000
                    % COMPUTE THE SURFACE VORTICIY                       00085100
                    %     USING A TAYLOR SERIES APPROXIMATION            00085200
     BEGIN                                                               00085300
     CREAL PRODUCT, PREFIX ;                                             00085400
                    % PARAMETERS REQUIRED, DTSI, PI                      00085500
     MODE := BCS ;                                                       00085600
     PRODUCT := PI * DTSI ;                                              00085700
     PREFIX := 1.0 / ( 2.0 * PRODUCT * PRODUCT ) ;                       00085800
     ZETA[0] := PREFIX * ( 8.0 * PSI[1] - PSI[2] ) ;                     00085900
     MODE := TRUE ;                                                      00086000
     END ;                                                               00086100
% ***************************************************************************00086200
%                                                                        00086300
                    PE REAL SUBROUTINE                                   00086400
INITIALIZE ( PCPOINT PSI, PCPOINT ZETA ) ;                               00086500
     BEGIN                                                               00086600
     CREAL EX, IX ;                                                      00086700
     CINT  JD, JDE ;                                                     00086800
                    % PARAMETERS REQUIRED, DX, IDX, LL1, S               00086900
                    %                                                    00087000
          BEGIN     % USE THE POTENTIAL FLOW SOLUTION AS THE             00087100
                    %    INITIAL STREAM FUNTION VALUES AND               00087200
                    %    ZERO THE VORTICITY NET                          00087300
          MODE := BCS ;                                                  00087400
          PSI[0] := 0.0 ;                                                00087500
          ZETA[0] := 0.0 ;                                               00087600
          EX := DX ;                                                     00087700
          IX := IDX ;                                                    00087800
          LOOP  JD  := 1,1, LL1   DO                                     00087900
               BEGIN                                                     00088000
               MODE := BCS ;                                             00088100
               ZETA[JD] := 0.0 ;                                         00088200
               PSI[JD] := 0.0 ;                                          00088300
               MODE := BC ;                                              00088400
               PSI[JD]  := (EX - IX)*S ;                                 00088500
               EX:= EX * DX ;                                            00088600
               IX:= IX * IDX ;                                           00088700
               JDE  := JD ;                                              00088800
               END ;                                                     00088900
          JD := JDE + 1 ;                                                00089000
          PSI[JD] := EX * S ;                                            00089100
          ZETA[JD] := 0.0 ;                                              00089200
          MODE := TRUE ;                                                 00089300
          END ;                                                          00089400
     PRINT1 (PSI,3) ;                                                    00089500
     POISSONDIRECT(PSI,ZETA,A,SS,NTSI,NETA,HX,HY,IS,PP);                 00089600
     SURFACEVORTICITY ( PSI, ZETA ) ;                                    00089700
     END;                                                                00089800
% ***************************************************************************00089900
%                                                                        00090000
                    PE REAL SUBROUTINE                                   00090100
PRINTPARAMETERS ;                                                        00090200
     BEGIN                                                               00090300
```

```
      MODE := TRUE ;                                                    00090400
      SIMWRITE (LINE,"REYNOLDS NUMBER", NREY,                           00090500
                     "NUMBER OF CIRCUMFERENTIAL MESH POINTS", NETA,     00090600
                     "NUMBER OF RADIAL MESH POINTS",NTSI,               00090700
                     "CIRCUMFERENTIAL INCREMENT SIZE",DETA,             00090800
                     "RADIAL INCREMENT SIZE",DTSI,                      00090900
                     "MAXIMUM CIRCUMFERENTIAL COORDINATE",ETAMAX,       00091000
                     "MAXIMUM RADIAL COORDINATE",TSIMAX,                00091100
                     "DIMENTIONLESS TIME INCREMENT",DT,                 00091200
                     "MAXIMUM DIMENTIONLESS TIME DESIRED",              00091300
                     TLMT, "NUMBER OF TIME STEPS BETWEEN PRINT-OUTS",   00091400
                     PINC, "BOOLEAN MASKS FOR FIELD WIDTH",             00091500
                     BCS,BCT) ;                                         00091600
      END ;                                                             00091700
%  ***************************************************************    00091800
%                                                                       00091900
                     PE REAL SUBROUTINE                                 00092000
OUTPUTRESULTS (CINT OUT ST, CREAL T) ;                                  00092100
      BEGIN                                                             00092200
      MODE := TRUE  ;                                                   00092300
      IF ST EQL PINC THEN                                               00092400
           BEGIN                                                        00092500
           SIMWRITE(LINE,"DIMENSIONLESS TIME",T ) ;                     00092600
           PRINT1 (ZETA,1) ;                                            00092700
           PRINT1 (PSI,2) ;                                             00092800
           ST := 0 ;                                                    00092900
$*         DISPLAY 0,1;                                                 00093000
           END ;                                                        00093100
      ST := ST + 1 ;                                                    00093200
      END ;                                                             00093300
%  ***************************************************************    00093400
%                                                                       00093500
                     PE REAL SUBROUTINE                                 00093600
MAIN ;                                                                  00093700
      BEGIN                                                             00093800
      LABEL REPEAT ;                                                    00093900
      CREAL  T ;                                                        00094000
                     % ACCUMULATED DIMENSIONLESS TIME                   00094100
      CINT ST ;                                                         00094200
                     % PRINTOUT PARAMETER                               00094300
      MODE := TRUE ;                                                    00094400
      T := 0.0 ;                                                        00094500
      ST := 1 ;                                                         00094600
      SYSTEMPARAMETERS ;                                                00094700
      COMPUTESYSTEMCONSTANTS ;                                          00094800
      PRINTPARAMETERS ;                                                 00094900
      SETPT1(A,SS,IS,PP,LL6 ,HX,HY) ;                                   00095000
      INITIALIZE ( PSI, ZETA ) ;                                        00095100
      OUTPUTRESULTS (ST,T) ;                                            00095200
                     % INITIALIZES THE STREAM FUNCTION AND VORTICITY    00095300
                     %    NETS, RELAXES THE PSI NET, AND COMPUTES AN    00095400
                     %    INITIAL SURFACE VORTICITY                     00095500
REPEAT:T := T + DT ;                                                    00095600
                     % INCREMENT TIME                                   00095700
      VORTICITY ( PSI, ZETA ) ;                                         00095800
                     % COMPUTE NEW VALUES OF VORTICITY BY ADI           00095900
      E := PI * DX ;                                                    00096000
      LOOP K := 1,1,LL1  DO                                             00096100
           BEGIN                                                        00096200
           ZETA[K] := ZETA[K] * E * E ;                                 00096300
```

```
            E := E * DX ;                                                    00096400
            END ;                                                            00096500
        POISSONDIRECT(PSI,ZETA,A,SS,NTSI,NETA,HX,HY,IS,PP);                  00096600
        E := PI * DX ;                                                       00096700
        LOOP K := 1,1,LL1  DO                                                00096800
            BEGIN                                                            00096900
            ZETA[K] := ZETA[K] / ( E * E ) ;                                 00097000
            E := E * DX ;                                                    00097100
            END ;                                                            00097200
                        % COMPUTE NEW VALUES OF THE STREAM FUNCTION           00097300
                        %    BY THE MODIFIED SUCCESSIVE OVERRELAXATION       00097400
                        %    METHOD                                          00097500
        IF ST NEQ PINC THEN                                                  00097600
            SIMWRITE (LINE, "NUMBER OF ITERATIONS = ", NITER );              00097700
        SURFACEVORTICITY ( PSI, ZETA ) ;                                     00097800
                        % ESTIMATES SURFACE VORTICITY BY REFERENCE TO         00097900
                        %    THE STREAM FUNCTION VALUES NEAR THE SURFACE     00098000
        OUTPUTRESULTS (ST,T) ;                                               00098100
        IF T LSS TLMT  THEN   GO TO REPEAT ;                                 00098200
        END ;                                                                00098300
%   ************************************************************************00098400
%   ************************************************************************00098500
%                                                                            00098600
        MAIN ;                                                               00098700
 FINI:  END .                                                                00098800
```

APPENDIX C

```
                    % CICHY/MASTER                                          00000100
BEGIN                                                                       00000200
REAL  DPSI, TIME2, GOP, YINF, TIMEI, REYN, DTIME,STARTTIME ;                00000300
INTEGER NXEND,NYEND,XD,YD,ZD,NITER,LNP,LNITER,POTENTIAL,                    00000400
                    SAVE,LNPP,RECTANGLE,NXLIM1,NXLIM2,                      00000500
                    NXLIM3,NYLIM1,NYLIM2,NYLIM3,MODS,                       00000600
                    PPZ,DSV,PZ,PP ;                                         00000700
FILE DATA(MAXRECSIZE=14,BLOCKSIZE=420,KIND=I,ACCESS=0,                      00000800
                    TITLE="CICHY/DATAMASTER.");                             00000900
READ(DATA,/,REYN,YINF,LNP,LNPP,LNITER,TIME2,DTIME,DPSI,TIMEI,               00001000
                    GOP,NITER,POTENTIAL,STARTTIME,SAVE,MODS,                00001100
                    DSV,PPZ,PZ,PP,NXEND,NYEND,RECTANGLE,NXLIMI,             00001200
                    NXLIM2,NXLIM3,NYLIMI,NYLIM2,NYLIM3 ) ;                  00001300
XD := NXEND ;                                                               00001400
YD := NYEND ;                                                               00001500
ZD := NYEND ;                                                               00001600
IF XD GTR YD THEN ZD := NXEND ;                                             00001700
     BEGIN                                                                  00001800
     DOUBLE ARRAY  ZETA[0:XD,0:YD] , PSI[0:XD,0:YD] ,                       00001900
                    EE[0:YD], ES[0:YD], RE[0:YD] ,RS[0:YD],                 00002000
                    RMOS[0:YD] , SMDT[0:YD] , F[0:ZD],                      00002100
                    D[0:ZD] , G[0:ZD], A[0:5,0:5], B[0:5,0:1],              00002200
                    RZETA[0:ZD],SLOPE[0:XD] ;                               00002300
     REAL ARRAY  ANGLE[0:XD]  ;                                             00002400
     ARRAY TITLEARRAY[0:17] ;                                               00002500
     DOUBLE  DPI, DP4, RXS, RYS, RDS4, RERX2, RERY2, WFBZ2,                 00002600
                    DELX, DELY, PI, DELXS, DELYS, PIS, RERXI,               00002700
                    RERX4, RERY1, RERY4, SIGMA, DELPI , RXYS ;              00002800
     REAL ETA, DELI, FK, DELTI, TYME, FI, WFBZ, WFP,                        00002900
                    PROTIM, IOTIM, ELAPTIM, TOTTIM  ;                       00003000
     INTEGER IT1, ONE, NP, K, J, ITZ, PROTIMIN, IOTIMIN,                    00003100
                    I, L, NXENDM, NXENDP, NYENDM, NYENDP,                   00003200
                    ELAPTIMIN, TOTTIMSTART, PROTIMOUT,                      00003300
                    IOTIMOUT, ELAPTIMOUT, PGNUM, HALF,                      00003400
                    NTEST, EVEN, NPRES, NIS, TOTTIMEND,INP,                 00003500
                    SECTION,NPP,NXFILS,NXFILF,NYFILS,                       00003600
                    NYFILF,IREC;                                            00003700
     POINTER FTITLE ;                                                       00003800
     ALPHA  PPRNT, ZPRNT, EPRNT, APRNT, BPRNT, NPRNT, IPRNT;                00003900
     LABEL  HOP999, HOPEND ;                                                00004000
     FILE LINE(KIND=I35,MYUSE=2,MAXRECSIZE=22,BUFFERS=2,                    00004100
                    INTMODE=3 ) ;                                           00004200
     FILE CARD(KIND=9,INTMODE=3,MAXRECSIZE=10 )  ;                          00004300
     FILE STOREP ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00004400
                    SAVEFACTOR=99,AREAS=20,AREASIZE=30,                     00004500
                    TITLE="CICHY/PSI.") ;                                   00004600
     FILE STOREZ ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00004700
                    SAVEFACTOR=99,AREAS=20,AREASIZE=30,                     00004800
                    TITLE="CICHY/ZETA.") ;                                  00004900
     FILE STOREC( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,              00005000
                    SAVEFACTOR=99,AREAS=1,AREASIZE=50,                      00005100
                    TITLE="CICHY/CONSTANTS.") ;                             00005200
     FILE SAVEP  ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00005300
                    ACCESS=I,SAVEFACTOR=99,                                 00005400
                    MAXRECSIZE=2*YD+2,TITLE="CICHY/SPSI.") ;                00005500
     FILE SAVEZ  ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00005600
                    ACCESS=I,SAVEFACTOR=99,                                 00005700
                    MAXRECSIZE=2*YD+2,TITLE="CICHY/SZETA.") ;               00005800
     FORMAT  FMTI (FI0.0,2I10,F10.0) ;  % I02                               00005900
     FORMAT  FMT2 (6F10.0) ;            % I03                               00006000
```

```
      FORMAT  FMT3 (2F10.3, 3I10 ) ;      % 105                          00006100
      FORMAT  FMT4 (X56, "PARAMETERS"/) ; % 107                          00006200
      FORMAT  FMT5 (X49,"REYN   =",F13.7/ X49,"YINF   =",F13.7/          00006300
                    X49,"LNP    =", I5  / X49,"LNPP   =", I5  /          00006400
                    X49,"LNITER =", I5  /                                00006500
                    X49,"TIME2  =",F13.7/ X49,"DTIME  =",F13.7/          00006600
                    X49,"DPSI   =",F13.7//X49,"TIME1  =",F13.7/          00006700
                    X49,"GOP    =",F13.7/ X49,"NITER  =", I5  /          00006800
                    X49,"POTENT =", I5  / X49,"WFP    =",F13.7//         00006900
                    X49,"NXEND  =", I5  / X49,"NYEND  =", I5  /          00007000
                    X49,"DELX   =",F13.7/ X49,"DELY   =",F13.7//         00007100
                    X49,"STTIME =",F13.7/ X49,"SAVE   =", I5  /          00007200
                    X49,"MODS   =", I5  //X49,"RECTANG=", I5  /          00007300
                    X49,"NXLIM1 =",F13.7/ X49,"NXLIM2 =",F13.7/          00007400
                    X49,"NXLIM3 =",F13.7/ X49,"NYLIM1 =",F13.7/          00007500
                    X49,"NYLIM2 =",F13.7/ X49,"NYLIM3 =",F13.7//         00007600
                    X49,"DSV    =", I5  / X49,"PPZ    =", I5  /          00007700
                    X49,"IREC   =", I5 // X49,"SECTION=", I5  ) ;        00007800
      FORMAT  FMT6 ("  ") ;                                              00007900
      FORMAT  FMT7 ("TIME =", F10.2) ;   % 108                           00008000
      FORMAT  FMT8 (10F8.4) ;                                            00008100
      FORMAT  FMT9 (/X3,"NUMBER OF ITERATIONS = ", I5 ) ;                00008200
      FORMAT  FMT16 (////) ;                                             00008300
      FORMAT  FMT75 ( //// X7,"CICHY",X108, "  PAGE ", I3// ) ;         00008400
      FORMAT  FMT90 ( // ) ;                                             00008500
%*****************************************************************************00008600
%                                                                        00008700
   PROCEDURE DATIME ;                                                    00008800
      BEGIN                                                              00008900
      VALUE ARRAY A(60"   SUN", 60"   MON", 60"  TUES",60"WEDNES",       00009000
                    60" THURS", 60"   FRI", 60" SATUR") ;                00009100
      INTEGER D,H,M,MIN,Q,P,Y , TIMEON ;                                 00009200
      LABEL OWT ;                                                        00009300
      TIMEON := TIME(1) ;                                                00009400
      Y := TIME(0) ;                                                     00009500
      D := Y.[17:6]*100 + 10*Y.[11:6] + Y.[5:6]  ;                       00009600
      Y := Y.[29:6]*10 + Y.[23:6] ;                                      00009700
      FOR H :=31,IF Y MOD 4=0 THEN 29                                    00009800
                    ELSE 28,31,30,31,30,31,31,30,31,30 DO                00009900
           IF D LEQ H THEN GO TO OWT                                     00010000
                    ELSE BEGIN D:=D-H;M:=M+1 END ;                       00010100
 OWT:  H:=TIMEON DIV 216000 ;                                            00010200
      MIN:=(TIMEON DIV 3600) MOD 60 ;                                    00010300
      IF M LSS 2 THEN BEGIN Q:=M+11;P:=Y-1 END                           00010400
                    ELSE BEGIN Q:=M-1;P:=Y END ;                         00010500
      M:=M+1 ;                                                           00010600
      FTITLE:=POINTER(TITLEARRAY,6) ;                                    00010700
      REPLACE FTITLE BY 6" " FOR 132  ;                                  00010800
      REPLACE FTITLE+85     BY       POINTER(A[((Q*26-2) DIV             00010900
                    10+D+P+P.[11:10]+1) MOD 7],6) FOR 6, 6"DAY, ",       00011000
                    M FOR 2 DIGITS,6"/", D FOR 2 DIGITS, 6"/",           00011100
                    Y FOR 2 DIGITS, 6", ",(12*REAL(Q:=H MOD 12=0)+Q)     00011200
                    FOR  2 DIGITS, 6":",MIN FOR 2 DIGITS,6" " ,          00011300
                    IF H GEQ 12 THEN 60"P"  ELSE 60"A" FOR 1, 6"M" ;     00011400
      END DATIME ;                                                       00011500
%*****************************************************************************00011600
%                                                                        00011700
   PROCEDURE  PRINT1( NETTYPE, NETNAME ) ;                               00011800
                    % PRINTOUT ROUTINE                                   00011900
      DOUBLE ARRAY  NETTYPE[*,*] ;                                       00012000
```

```
        ALPHA  NETNAME ;                                                    00012100
        BEGIN                                                               00012200
        ALPHA  DESIG ;                                                      00012300
        INTEGER NC, II, M1, M2 , KK , NXMARK, NXLIM, SECHAF ;               00012400
        LABEL HOP512, HOP503  , HOP504 ;                                    00012500
        FORMAT FMT10( X20,"NITER ="I5,X23,  "SECTION NUMBER  ", I3, A4,     00012600
                      X5, A4, X19, "TIME =", F7.3/ ) ;                      00012700
        FORMAT FMT11( 11E12.4 ) ;                                           00012800
        NC := NYEND / 11  + 1 ;                                             00012900
        FOR  II := 1 STEP 1 UNTIL NC   DO                                   00013000
             BEGIN                                                          00013100
             SECHAF := 0 ;                                                  00013200
             M1 := 11 * II - 10 ;                                           00013300
             IF (M1 - NYEND) GTR 0.0  THEN GO TO HOP512 ;                   00013400
             M2 := M1 + 10 ;                                                00013500
             IF (M2 - NYEND) LEQ 0.0  THEN GO TO HOP504 ;                   00013600
             M2 := NYEND ;                                                  00013700
HOP504:      IF HALF=1 THEN BEGIN NXMARK:=NXEND/2.0; NXLIM:= 1 ;            00013800
                       DESIG := APRNT; END                                  00013900
                       ELSE BEGIN NXMARK:=NXEND; NXLIM:= 1 ;                00014000
                       DESIG := NPRNT ; END ;                               00014100
HOP503:      WRITE (LINE [SKIP 1 ] ) ;                                      00014200
             PGNUM := PGNUM + 1 ;                                           00014300
             WRITE (LINE, FMT75, PGNUM ) ;                                  00014400
             WRITE ( LINE, 16 , TITLEARRAY[*] ) ;                           00014500
             WRITE (LINE, FMT90 ) ;                                         00014600
             WRITE (LINE,FMT10, NITER, II, DESIG, NETNAME, TIME1 ) ;        00014700
             FOR  I := NXLIM STEP 1 UNTIL NXMARK  DO                        00014800
                  BEGIN                                                     00014900
                  WRITE(LINE, FMT11, FOR J := M1 STEP 1 UNTIL M2 DO         00015000
                       NETTYPE[I,J]) ;                                      00015100
                  END ;                                                     00015200
             IF SECHAF = 1 THEN GO TO HOP512 ;                              00015300
             IF HALF=1 THEN BEGIN SECHAF:=1; NXLIM:= NXMARK + 1;            00015400
                       NXMARK:=NXEND;  DESIG := BPRNT ;                     00015500
                       GO TO HOP503; END ;                                  00015600
HOP512:      END ;                                                          00015700
        END ;                                                               00015800
%***********************************************************************  00015900
%                                                                           00016000
   PROCEDURE TYMIN;                                                         00016100
                       % INITIALIZE THE TIMING ROUTINE                      00016200
        BEGIN                                                               00016300
        PROTIMIN:= TIME(2) ;                                                00016400
        IOTIMIN := TIME(3) ;                                                00016500
        ELAPTIMIN := TIME(1) ;                                              00016600
        END ;                                                               00016700
%***********************************************************************  00016800
%                                                                           00016900
   PROCEDURE TYMOUT ;                                                       00017000
                       % OUTPUT TIMING INFORMATION                          00017100
        BEGIN                                                               00017200
        FORMAT FMT50 (X4,"****","PROCESS TIME (SEC) =", F6.2, X2, "**",     00017300
                       "I/O TIME (SEC) =",F5.2, X2, "**",                   00017400
                       "ELAPSED TIME (SEC) =", F6.2, X2, "**",              00017500
                       "TOTAL ELAPSED TIME (MINUTES) =", F5.2 ) ;           00017600
        FORMAT FMT51 (//////////);                                          00017700
        PROTIMOUT := TIME(2) ;                                              00017800
        IOTIMOUT := TIME(3) ;                                               00017900
        ELAPTIMOUT := TIME(1) ;                                             00018000
```

```
        PROTIM := (PROTIMOUT - PROTIMIN)/60 ;                                   00018100
        IOTIM := (IOTIMOUT - IOTIMIN)/60 ;                                      00018200
        ELAPTIM := (ELAPTIMOUT - ELAPTIMIN)/60 ;                                00018300
        TOTTIMEND := TIME(1) ;                                                  00018400
        TOTTIM  := (TOTTIMEND - TOTTIMSTART)/3600 ;                             00018500
        WRITE(LINE,FMT51) ;                                                     00018600
        WRITE( LINE, FMT50, PROTIM, IOTIM, ELAPTIM, TOTTIM);                    00018700
        END ;                                                                   00018800
%***************************************************************************00018900
%                                                                               00019000
   PROCEDURE  SAVEPSIZETA ;                                                     00019100
        BEGIN                                                                   00019200
        INTEGER RECORD ;                                                        00019300
        IF IREC=1 THEN                                                          00019400
             BEGIN                                                              00019500
             SAVEP.AREASIZE := XD ;                                             00019600
             SAVEP.AREAS := 100 ;                                               00019700
                                                                                00019800
             WRITE(SAVEP[0],*,REYN,YINF,                                        00019900
                       DTIME,DPSI,POTENTIAL,                                    00020000
                       WFP,NXEND,NYEND,DELX,DELY,STARTTIME,                     00020100
                       MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,                     00020200
                       NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;                    00020300
             SAVEZ.AREASIZE := XD ;                                             00020400
             SAVEZ.AREAS := 100 ;                                               00020500
             WRITE(SAVEZ[0],*,REYN,YINF,                                        00020600
                       DTIME,DPSI,POTENTIAL,                                    00020700
                       WFP,NXEND,NYEND,DELX,DELY,STARTTIME,                     00020800
                       MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,                     00020900
                       NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;                    00021000
             LOCK(SAVEP) ;                                                      00021100
             LOCK(SAVEZ) ;                                                      00021200
             SAVEP.AREASIZE := 0 ;                                              00021300
             SAVEP.AREAS := 0 ;                                                 00021400
             SAVEZ.AREASIZE := 0 ;                                              00021500
             SAVEZ.AREAS := 0 ;                                                 00021600
             END ;                                                              00021700
        I := 1 ;                                                                00021800
        FOR RECORD:=IREC STEP 1 UNTIL XD-1+IREC DO                              00021900
             BEGIN                                                              00022000
             WRITE(SAVEP[RECORD],2*YD+2,PSI[I,*]);                              00022100
             WRITE(SAVEZ[RECORD],2*YD+2,ZETA[I,*]);                             00022200
             I := I + 1 ;                                                       00022300
             END ;                                                              00022400
        IREC := IREC + XD ;                                                     00022500
        CLOSE(SAVEP) ;                                                          00022600
        CLOSE(SAVEZ) ;                                                          00022700
        END ;                                                                   00022800
%***************************************************************************00022900
%                                                                               00023000
   PROCEDURE STOREPSIZETA ;                                                     00023100
        BEGIN                                                                   00023200
        STOREP.MAXRECSIZE:=2*YD+2 ;                                             00023300
        STOREP.AREAS:=20 ;                                                      00023400
        STOREP.AREASIZE:=30 ;                                                   00023500
        FOR I:=1 STEP 1 UNTIL XD DO WRITE(STOREP,2*YD+2,PSI[I,*]) ;             00023600
        LOCK(STOREP) ;                                                          00023700
```

```
        STOREZ.MAXRECSIZE:=2*YD+2 ;
        STOREZ.AREAS:=20 ;
        STOREZ.AREASIZE:=30 ;
        FOR I:=1 STEP 1 UNTIL XD DO WRITE(STOREZ,2*YD+2,ZETA[I,*]) ;
        LOCK(STOREZ) ;
        END ;
%*************************************************************************
%
   PROCEDURE STORECONSTANTS ;
        BEGIN
        STOREC.MAXRECSIZE := 50 ;
        STOREC.AREAS := 1 ;
        STOREC.AREASIZE := 50 ;
        WRITE(STOREC, * , REYN,YINF,DTIME,DPSI,TIME1,GOP,
                      NITER,NXEND,NYEND,SECTION,RECTANGLE,
                      NXLIM1,NXLIM2,NXLIM3,NYLIM1,NYLIM2,
                      NYLIM3,MODS,DSV,IREC,PPZ,NP,NPP,PZ,PP ) ;
        LOCK(STOREC) ;
        END ;
%*************************************************************************
%
   PROCEDURE  SETUPGET ;
        BEGIN
        IF SECTION = 1 THEN
             BEGIN
             READ (SAVEP[0],*,REYN,YINF,
                       DTIME,DPSI,POTENTIAL,
                       WFP,NXEND,NYEND,DELX,DELY,STARTTIME,
                       MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,
                       NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;
             READ (SAVEZ[0],*,REYN,YINF,
                       DTIME,DPSI,POTENTIAL,
                       WFP,NXEND,NYEND,DELX,DELY,STARTTIME,
                       MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,
                       NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;
             END ;
        END ;
%*************************************************************************
%
   PROCEDURE GETPSIZETA ;
        BEGIN
        INTEGER RECORD ;
        I := 1 ;
        FOR RECORD := IREC STEP 1 UNTIL XD-1+IREC  DO
             BEGIN
             READ(SAVEP[RECORD],2*YD+2,PSI[I,*]);
             READ(SAVEZ[RECORD],2*YD+2,ZETA[I,*]);
             I := I + 1 ;
             END ;
        IREC := IREC + XD ;
        END ;
%*************************************************************************
%
   PROCEDURE FETCHPSIZETA ;
        BEGIN
        STOREP.AREAS:=0 ;
        STOREP.AREASIZE:=0 ;
        STOREZ.AREAS:=0 ;
        STOREZ.AREASIZE:=0 ;
        IF STOREP.PRESENT THEN
```

```
            BEGIN                                                          00029800
            LABEL EOF1,EOF2 ;                                              00029900
            FOR I:= 1 STEP 1 UNTIL XD DO                                   00030000
                      READ(STOREP,2*YD+2,PSI[I,*])[EOF1] ;                 00030100
 EOF1:      FOR I:= 1 STEP 1 UNTIL XD DO                                   00030200
                      READ(STOREZ,2*YD+2,ZETA[I,*])[EOF2] ;                00030300
 EOF2:      I:= I-1;                                                       00030400
            J:= (STOREP.MAXRECSIZE) DIV 2 ;                                00030500
            CLOSE (STOREP) ;                                               00030600
            CLOSE (STOREZ) ;                                               00030700
            END ;                                                          00030800
       END ;                                                               00030900
%**********************************************************************00031000
%                                                                          00031100
   PROCEDURE FETCHCONSTANTS ;                                              00031200
       BEGIN                                                               00031300
       STOREC.AREAS := 0 ;                                                 00031400
       STOREC.AREASIZE :=  0 ;                                             00031500
       IF STOREC.PRESENT THEN                                              00031600
            BEGIN                                                          00031700
            READ(STOREC, * , REYN,YINF,DTIME,DPSI,TIME1,GOP,               00031800
                     NITER,NXEND,NYEND,SECTION,RECTANGLE,                  00031900
                     NXLIM1,NXLIM2,NXLIM3,NYLIM1,NYLIM2,                   00032000
                     NYLIM3,MODS,DSV,IREC,PPZ,NP,NPP,PZ,PP ) ;             00032100
            CLOSE(STOREC) ;                                                00032200
            END ;                                                          00032300
       END ;                                                               00032400
%**********************************************************************00032500
%                                                                          00032600
   PROCEDURE INITIALIZEPROGRAM ;                                           00032700
       BEGIN                                                               00032800
       LABEL SKIP ;                                                        00032900
       TOTTIMSTART := TIME(1) ;                                            00033000
       TYMIN ;                                                             00033100
       DATIME ;                                                            00033200
       SECTION := 0 ;                                                      00033300
       NP := 0 ;                                                           00033400
       NPP := 0 ;                                                          00033500
       INP := 4 ;                                                          00033600
       IF PPZ=1 AND NITER=0 THEN GO TO SKIP ;                              00033700
       FETCHCONSTANTS ;                                                    00033800
 SKIP: SECTION := SECTION + 1 ;                                            00033900
       IF SECTION=1 THEN IREC:=1 ;                                         00034000
       PGNUM := 0 ;                                                        00034100
       IT1 := TIME(2) ;                                                    00034200
       LNITER := LNITER + NITER ;                                          00034300
       REPLACE FTITLE+15 BY                                                00034400
            6"CICHY/MASTER FLOW AROUND A CYLINDER - REVISED 05/16/72";     00034500
       END ;                                                               00034600
%**********************************************************************00034700
%                                                                          00034800
   PROCEDURE COMPUTECONSTANTS ;                                            00034900
       BEGIN                                                               00035000
       EPRNT :=6"ERRR" ;                                                   00035100
       PPRNT :=6"PSI " ;                                                   00035200
       ZPRNT :=6"ZETA" ;                                                   00035300
       APRNT :=6"A   " ;                                                   00035400
       BPRNT :=6"B   " ;                                                   00035500
       NPRNT :=6"    " ;                                                   00035600
       IPRNT :=6"IPSI" ;                                                   00035700
```

```
                % IF NXEND EXCEEDS 50 THE OUTPUT IS SPLIT          00035800
                %    AND PUT ON TWO PAGES.                        00035900
IF NXEND GTR 50 THEN HALF := 1 ELSE HALF := 0 ;                   00036000
NTEST := NXEND / 2 ;                                              00036100
IF ( 2 * NTEST - NXEND) = 0 THEN EVEN := 1 ELSE EVEN := 0 ;       00036200
NXENDM := NXEND - 1 ;                                             00036300
NYENDM := NYEND - 1 ;                                             00036400
NXENDP := NXEND +1 ;                                              00036500
NYENDP := NYEND +1 ;                                              00036600
IF EVEN = 1 THEN NPRES := NXEND/2 ELSE NPRES := NXENDM/2 + 1 ;    00036700
DELX := 1.0 / NXENDM ;                                            00036800
DELY := YINF / NYENDM ;                                           00036900
DEL1 := 180.0 * DELX ;                                            00037000
ONE  := 1 ;                                                       00037100
PI := 4 * DARCTAN(ONE)  ;                                         00037200
PIS := PI * PI ;                                                  00037300
DELPI := 2.0 * PI * DELY ;                                        00037400
FOR  K := 1 STEP 1 UNTIL NYENDM  DO                               00037500
     BEGIN                                                        00037600
     FK := K - 1 ;                                                00037700
     SIGMA := FK * DELPI ;                                        00037800
     RMOS[K]:= PIS * DEXP(SIGMA) ;                                00037900
     END;                                                         00038000
DELXS := DELX * DELX ;                                            00038100
DELYS := DELY * DELY ;                                            00038200
WFBZ2 :=  1.0 / (2.0 *  PIS * DELYS) ;                            00038300
RXS := 1.0 / DELXS ;                                              00038400
RYS := 1.0 / DELYS ;                                              00038500
RXYS := 1.0 / (2.0 * (RXS + RYS) ) ;                              00038600
RDS4 := 0.25 / (DELX * DELY) ;                                    00038700
RERX1 := RXS / REYN ;                                             00038800
RERX2 := 2.0 * RERX1 ;                                            00038900
RERX4 := 4.0 * RERX1 ;                                            00039000
RERY1 := RYS / REYN ;                                             00039100
RERY2 := 2.0 * RERY1 ;                                            00039200
RERY4 := 4.0 * RERY1 ;                                            00039300
                % COMPUTE OVERRELAXATION PARAMETER                 00039400
     BEGIN                                                        00039500
     REAL DR,DRE ;                                                00039600
     DR := 0.5 * LN(0.5*((1.0/NXEND)**2 + (1.0/NYEND)**2));       00039700
     DRE := EXP(DR);                                              00039800
     WFP := 2.0 / (1.0 + PI * DRE) ;                              00039900
     END ;                                                        00040000
DP1 := 1.0 - WFP ;                                                00040100
DP4 := WFP * RXYS ;                                               00040200
FOR  J := 1  STEP 1 UNTIL NPRES  DO                               00040300
     BEGIN                                                        00040400
     FI := 2 * J - 2 ;                                            00040500
     ANGLE[J] := DEL1 * FI ;                                      00040600
     END ;                                                        00040700
IF EVEN = 1 THEN ANGLE[ NPRES + 1 ] := DEL1 * (FI + 1) ;          00040800
DELTI := DTIME / 2.0 ;                                            00040900
FOR  J := 2 STEP 1 UNTIL NYENDM  DO                               00041000
     BEGIN                                                        00041100
     SMDT[J] := RMOS[J] / DELTI ;                                 00041200
     EE[J] := SMDT[J] + RERX4 ;                                   00041300
     ES[J] := SMDT[J] + RERY4 ;                                   00041400
     RE[J] := SMDT[J] - RERY4 ;                                   00041500
     RS[J] := SMDT[J] - RERX4 ;                                   00041600
     END ;                                                        00041700
```

```
        END ;                                                                   00041800
%***************************************************************************00041900
%                                                                               00042000
   PROCEDURE NETA ;                                                             00042100
        BEGIN                                                                   00042200
        NXFILS := 2 ;                                                           00042300
        NXFILF := NXENDM ;                                                      00042400
        NYFILS := 2 ;                                                           00042500
        NYFILF := NYLIM1-1 ;                                                    00042600
        END ;                                                                   00042700
%***************************************************************************00042800
%                                                                               00042900
   PROCEDURE NETB ;                                                             00043000
        BEGIN                                                                   00043100
        NXFILS := NXLIM1+1 ;                                                    00043200
        NXFILF := NXENDM ;                                                      00043300
        NYFILS := NYLIM1 ;                                                      00043400
        NYFILF := NYLIM2-1 ;                                                    00043500
        END ;                                                                   00043600
%***************************************************************************00043700
%                                                                               00043800
   PROCEDURE NETC ;                                                             00043900
        BEGIN                                                                   00044000
        NXFILS := NXLIM2+1 ;                                                    00044100
        NXFILF := NXENDM ;                                                      00044200
        NYFILS := NYLIM2 ;                                                      00044300
        NYFILF := NYENDM ;                                                      00044400
        END ;                                                                   00044500
%***************************************************************************00044600
%                                                                               00044700
   PROCEDURE NETD ;                                                             00044800
        BEGIN                                                                   00044900
        NXFILS := 2 ;                                                           00045000
        NXFILF := NXLIM1 ;                                                      00045100
        NYFILS := 2 ;                                                           00045200
        NYFILF := NYLIM1-1 ;                                                    00045300
        END ;                                                                   00045400
%***************************************************************************00045500
%                                                                               00045600
   PROCEDURE NETE ;                                                             00045700
        BEGIN                                                                   00045800
        NXFILS := NXLIM1+1 ;                                                    00045900
        NXFILF := NXLIM2 ;                                                      00046000
        NYFILS := 2 ;                                                           00046100
        NYFILF := NYLIM2-1 ;                                                    00046200
        END ;                                                                   00046300
%***************************************************************************00046400
%                                                                               00046500
   PROCEDURE NETF ;                                                             00046600
        BEGIN                                                                   00046700
        NXFILS := NXLIM2+1 ;                                                    00046800
        NXFILF := NXENDM ;                                                      00046900
        NYFILS := 2 ;                                                           00047000
        NYFILF := NYENDM ;                                                      00047100
        END ;                                                                   00047200
%***************************************************************************00047300
%                                                                               00047400
   PROCEDURE NETG ;                                                             00047500
        BEGIN                                                                   00047600
        NXFILS := 2 ;                                                           00047700
```

```
        NXFILF := NXENDM ;                                                  00047800
        NYFILS := 2 ;                                                       00047900
        NYFILF := NYENDM ;                                                  00048000
        END ;                                                               00048100
%***************************************************************************00048200
%                                                                           00048300
   PROCEDURE THOMAS ( NFILS,NFILF, E , TZETA ) ;                            00048400
                      % THE THOMAS METHOD FOR INVERTING                     00048500
                      %    TRIDINGONAL MATRICIES                           00048600
        DOUBLE ARRAY  E[*] , TZETA[*] ;                                     00048700
        INTEGER NFILS,NFILF ;                                               00048800
        BEGIN                                                               00048900
        DOUBLE ARRAY WR[0:ZD ], VR[0:ZD ], RR[0:ZD ]  ;                     00049000
        INTEGER I, J, K ,L  , K2 ;                                          00049100
        WR[NFILS] := D[NFILS] / E[NFILS] ;                                  00049200
        VR[NFILS] := G[NFILS] / E[NFILS] ;                                  00049300
        FOR K := NFILS+1 STEP 1 UNTIL NFILF   DO                            00049400
             BEGIN                                                          00049500
             RR[K] := E[K] - F[K] * WR[K-1]   ;                             00049600
             WR[K] := D[K] / RR[K] ;                                        00049700
             END ;                                                          00049800
        FOR K := NFILS+1 STEP 1 UNTIL NFILF  DO                             00049900
             BEGIN                                                          00050000
             VR[K] := (G[K] - F[K]*VR[K-1] ) / RR[K] ;                      00050100
             END ;                                                          00050200
        L := NFILF ;                                                        00050300
        TZETA[L] := VR[L] ;                                                 00050400
        K2 := NFILF -2 ;                                                    00050500
        FOR  K := NFILS-1 STEP 1 UNTIL K2  DO                               00050600
             BEGIN                                                          00050700
             L := NFILF - K ;                                               00050800
             TZETA[L] := VR[L] - WR[L] * TZETA[L+1]  ;                      00050900
             END ;                                                          00051000
        END ;                                                               00051100
%***************************************************************************00051200
%                                                                           00051300
   PROCEDURE VORTE      ;                                                   00051400
                      % FIRST HALF OF ADI                                   00051500
        BEGIN                                                               00051600
        DOUBLE ARRAY TE[0:XD ] , TZETA[0:XD] ;                              00051700
        DOUBLE PFR1, PFR2, Q, S  ;                                          00051800
        INTEGER I, J, K ,L  ;                                               00051900
        FOR  I := NXFILS STEP 1 UNTIL NXFILF  DO                            00052000
             BEGIN                                                          00052100
             TZETA[I] := ZETA[I,NYFILS-1] ;                                 00052200
             END ;                                                          00052300
        FOR  J := NYFILS STEP 1 UNTIL NYFILF  DO                            00052400
             BEGIN                                                          00052500
             FOR I := NXFILS STEP 1 UNTIL NXFILF  DO                        00052600
                  BEGIN                                                     00052700
                  PFR1 := -RDS4 * ( PSI[I+1,J] - PSI[I-1,J] ) ;             00052800
                  PFR2 := -RDS4 * ( PSI[I,J+1] - PSI[I,J-1] ) ;             00052900
                  F[I] := PFR2 - RERX2 ;                                    00053000
                  TE[I] := EE[J] ;                                          00053100
                  D[I]  := - PFR2 - RERX2 ;                                 00053200
                  Q := RERY2 + PFR1 ;                                       00053300
                  S := RERY2 - PFR1 ;                                       00053400
                  G[I] := Q * RZETA[I]    + RE[J] * ZETA[I,J]               00053500
                       + S * ZETA[I,J+1] ;                                  00053600
                  RZETA[I] := ZETA[I,J] ;                                   00053700
```

```
                ZETA[I,J-1] := TZETA[I] ;                                    00053800
                END  ;                                                       00053900
           THOMAS ( NXFILS, NXFILF,TE , TZETA ) ;                            00054000
           END ;                                                             00054100
      FOR  I:= NXFILS STEP 1 UNTIL NXFILF  DO                                00054200
           BEGIN                                                             00054300
           ZETA[I,NYFILF] := TZETA[I] ;                                      00054400
           END ;                                                             00054500
      END;                                                                   00054600
%************************************************************************00054700
%                                                                           00054800
   PROCEDURE VORTS    ;                                                     00054900
                     % SECOND HALF OF ADI                                   00055000
      BEGIN                                                                  00055100
      DOUBLE ARRAY TZETA[0:YD] ;                                             00055200
      DOUBLE PFR1, PFR2, Q, S  ;                                             00055300
      INTEGER I, J, K ,L  ;                                                  00055400
      FOR  J := NYFILS STEP 1 UNTIL NYFILF   DO                              00055500
           BEGIN                                                             00055600
           TZETA[J] := ZETA[NXFILS-1,J] ;                                    00055700
           END;                                                              00055800
      FOR  I := NXFILS STEP 1 UNTIL NXFILF  DO                               00055900
           BEGIN                                                             00056000
           FOR J := NYFILS STEP 1 UNTIL NYFILF  DO                           00056100
                BEGIN                                                        00056200
                PFR1 := -RDS4 * (PSI[I+1,J] - PSI[I-1,J] ) ;                 00056300
                PFR2 := -RDS4 * (PSI[I,J+1] - PSI[1,J-1] ) ;                 00056400
                F[J] := -PFR1 - RERY2 ;                                      00056500
                D[J] :=  PFR1 - RERY2 ;                                      00056600
                Q := RERX2 - PFR2 ;                                          00056700
                S := RERX2 + PFR2 ;                                          00056800
                G[J] := Q * RZETA[J]    + RS[J] * ZETA[I,J]                  00056900
                     + S * ZETA[I+1,J] ;                                     00057000
                RZETA[J] := ZETA[I,J] ;                                      00057100
                ZETA[I-1,J] := TZETA[J] ;                                    00057200
                END ;                                                        00057300
           G[NXFILS] := G[NXFILS] - F[NXFILS] * ZETA[I,NXFILS-1] ;           00057400
           THOMAS  ( NYFILS, NYFILF, ES , TZETA ) ;                          00057500
           END ;                                                             00057600
      FOR  J := NYFILS STEP 1 UNTIL NYFILF  DO                               00057700
           BEGIN                                                             00057800
           ZETA[NXFILF,J] := TZETA[J] ;                                      00057900
           END ;                                                             00058000
      END ;                                                                  00058100
%************************************************************************00058200
   PROCEDURE ADI ;                                                           00058300
      BEGIN                                                                  00058400
      LABEL L2 ;                                                             00058500
      IF RECTANGLE EQL 1 THEN                                                00058600
           BEGIN                                                             00058700
           NETG ;                                                            00058800
           FOR I := NXFILS STEP 1 UNTIL NXFILF DO                            00058900
                BEGIN                                                        00059000
                RZETA[I] := ZETA[I,NYFILS-1] ;                               00059100
                END ;                                                        00059200
           VORTE ;                                                           00059300
           FOR J := NYFILS STEP 1 UNTIL NYFILF DO                            00059400
                BEGIN                                                        00059500
                RZETA[J] := ZETA[NXFILS-1,J] ;                               00059600
                END ;                                                        00059700
```

```
          VORTS ;                                                    00059800
          GO TO L2 ;                                                 00059900
          END ;                                                      00060000
     NETA ;                                                          00060100
     FOR I := NXFILS STEP 1 UNTIL NXFILF DO                          00060200
          BEGIN                                                      00060300
          RZETA[I] := ZETA[I,NYFILS-1];                              00060400
          END ;                                                      00060500
     VORTE ;                                                         00060600
     NETB ;                                                          00060700
     VORTE ;                                                         00060800
     NETC ;                                                          00060900
     VORTE;                                                          00061000
     NETD ;                                                          00061100
     FOR J := NYFILS STEP 1 UNTIL NYFILF DO                          00061200
          BEGIN                                                      00061300
          RZETA[J] := ZETA[NXFILS-1,J] ;                             00061400
          END ;                                                      00061500
     VORTS ;                                                         00061600
     NETE ;                                                          00061700
     VORTS ;                                                         00061800
     NETF ;                                                          00061900
     VORTS ;                                                         00062000
L2:  END ;                                                           00062100
%****************************************************************** 00062200
%                                                                    00062300
   PROCEDURE MODSTREAM ;                                             00062400
     BEGIN                                                           00062500
     DEFINE NEWPSI =                                                 00062600
          BEGIN                                                      00062700
          TPSI :=DP1 * PSI[I,J] + DP4 * ( RXS * (PSI[I-1,J]          00062800
                + PSI[I+1,J]) + RYS * (PSI[I,J-1] + PSI[I,J+1])      00062900
                - RMOS[J] * ZETA[I,J] ) ;                            00063000
          TP := DABS( TPSI - PSI[I,J] ) ;                            00063100
          IF TP GTR DPSI THEN GOP := 2.0 ;                           00063200
          PSI[I,J] := TPSI ;                                         00063300
          END # ;                                                    00063400
     INTEGER     C, PASS, SW, I, J;                                  00063500
     DOUBLE TPSI ;                                                   00063600
     REAL TP ;                                                       00063700
     LABEL HOP444, HOP405, HOP490, L1, L2 ;                          00063800
     NIS := 1 ;                                                      00063900
HOP405:GOP:=1.0 ;                                                    00064000
     SW := 0 ;                                                       00064100
     PASS := 1 ;                                                     00064200
HOP444:C := INP ;                                                    00064300
L1:  CASE C OF BEGIN NETD;NETE;NETF;GO TO L2;NETG;GO TO L2;END ;     00064400
     C := C + 1 ;                                                    00064500
     FOR I := NXFILS STEP 1 UNTIL NXFILF DO                          00064600
          BEGIN                                                      00064700
          FOR J := SW + NYFILS STEP 2 UNTIL NYFILF DO NEWPSI ;       00064800
          IF SW = 0 THEN SW:=1 ELSE SW:=0 ;                          00064900
          END ;                                                      00065000
     GO TO L1 ;                                                      00065100
L2:  IF PASS=1 THEN BEGIN PASS:=2; SW:=1; GO TO HOP444; END ;        00065200
     IF GOP LSS 1.5 THEN GO TO HOP490  ;                             00065300
     NIS := NIS + 1 ;                                                00065400
     ITZ := TIME(2)  ; % ELAPSED TIME OF JOB IN 1/60 SECS            00065500
     TYME := ITZ / 60 ;%TIME IN SECONDS                              00065600
     IF TYME LSS TIME2 THEN GO TO HOP405 ;                           00065700
```

```
 HOP490:END ;
%*********************************************************************
%
   PROCEDURE SWEEP      ;
        BEGIN
        DEFINE NEWPSI =
             BEGIN
             TPSI :=DP1 * PSI[I,J] + DP4 * ( RXS * (PSI[I-1,J]
                       + PSI[I+1,J]) + RYS * (PSI[I,J-1] + PSI[1,J+1])
                       - RMOS[J] * ZETA[I,J] ) ;
             TP := DABS( TPSI - PSI[I,J] ) ;
             IF TP GTR DPSI THEN GOP := 2.0 ;
             PSI[I,J] := TPSI ;
             END # ;
        INTEGER         I, J;
        DOUBLE TPSI ;
        REAL TP ;
        FOR I := NXFILS STEP 1 UNTIL NXFILF  DO
             BEGIN
             FOR J := NYFILS STEP 1 UNTIL NYFILF DO NEWPSI ;
             END;
        END ;
%*********************************************************************
%
   PROCEDURE STREAM ;
        BEGIN
        LABEL HOP405, HOP490, L1, L2 ;
        INTEGER C ;
        IF MODS = 1 THEN BEGIN MODSTREAM ; GO TO HOP490 ; END ;
        NIS := 1 ;
 HOP405:GOP := 1.0 ;
        C := INP ;
 L1:    CASE C OF BEGIN NETD;NETE;NETF;GO TO L2;NETG;GO TO L2;END ;
        SWEEP ;
        C := C + 1 ;
        GO TO L1 ;
 L2:    IF GOP LSS 1.5 THEN GO TO HOP490 ;
        NIS := NIS + 1 ;
        ITZ := TIME(2) ; % ELAPSED TIME OF JOB IN 1/60 SECS
        TYME := ITZ / 60 ; % TIME IN SECONDS
        IF TYME LSS TIME2 THEN GO TO HOP405 ;
 HOP490:END ;
%*********************************************************************
%
   PROCEDURE  SURFACEVORTICITY ;
                       % COMPUTE THE SURFACE VORTICITY
        BEGIN
        INTEGER I, J, K ,L  ;
        FOR  I := 2 STEP 1 UNTIL NXENDM   DO
             BEGIN
             ZETA[I,1]:=(8.0 * PSI[I,2] - PSI[I,3] ) * WFBZ2 ;
             END ;
        END ;
%*********************************************************************
%
   PROCEDURE SHORTTIMESOLUTION ( T ) ;
        DOUBLE T ;
        BEGIN
        DOUBLE ARRAY S[0:XD] , C[0:XD], FDZT[0:XD] ;
        DOUBLE  VAR1,VAR2,VAR3,VAR4,VAR5,VAR6,ALF,CST1,CST2,CST3,CST4,
```

```
                    CST5,CST6,CST7,CST8,CST9,CST10,CST11,CST12,      00071800
                    CST13,CST14,CST15,CST16,CST17,CST18,CST19,       00071900
                    CST20,CST21,CST22,CST23,CST24,VAR7,CST25,CST26,  00072000
                    INT , TSI, ETA , DTSI, LEVEL,                    00072100
                    VAR8,VAR9,VAR12,VAR13,                           00072200
                    VAR10, CST27, CST28, CST29, CST30, VAR11,        00072300
                    VAR14,PITSI,VAR15,VAR16,VAR17,VAR18,             00072400
                    VAR19,VAR20,VAR21,VAR22,VAR23,CST31,VAR24 ;      00072500
      INTEGER CC ;                                                   00072600
      LABEL L1,L2 ;                                                  00072700
      CST1 := DSQRT(PI) ;                                            00072800
      CST2 := 1.0/CST1 ;                                             00072900
      CST3 := 1.0 / PI ;                                             00073000
      CST4 := PI * PI ;                                              00073100
      CST5 := CST3 * CST3 ;                                          00073200
      CST6 := DSQRT(2.0);                                            00073300
      CST7 := 1.0 / CST6 ;                                           00073400
      CST8 := 1.0 / 3.0 ;                                            00073500
      CST9 := 4.0 * CST2 ;                                           00073600
      CST10 := 0.25 * CST1 ;                                         00073700
      CST11 := 1.5 * CST1 ;                                          00073800
      CST12 := CST8 * CST3 ;                                         00073900
      CST13 := 0.5 - 2.0 * CST12 ;                                   00074000
      CST14 := 2.0 * CST8 * CST2 ;                                   00074100
      CST15 := CST14 * CST14 ;                                       00074200
      CST16 := 1.0 + CST15 ;                                         00074300
      CST17 := CST2 * CST16 ;                                        00074400
      CST18 := 11.0 * CST2 / 6.0 ;                                   00074500
      CST19 := CST2 * ( CST15 - 1.5 ) ;                              00074600
      CST20 := 8.0 * CST8 * CST7 * CST2 ;                            00074700
      CST21 := CST2 * ( 8.0 * CST8 * CST7 - CST15 - 1.0 ) ;          00074800
      CST22 := 2.0 * T / REYN ;                                      00074900
      CST23 := DSQRT( CST22 ) ;                                      00075000
      CST24 := 1.0 / CST23 ;                                         00075100
      CST25 := 0.5 * CST24 ;                                         00075200
      CST26 := 8.0 * CST8 * CST7 ;                                   00075300
      CST27 := REYN /( 8.0 * T ) ;                                   00075400
      CST28 := 4.0 * CST22 ;                                         00075500
      CST29 := DSQRT(CST28) ;                                        00075600
      CST30 := 4.0 * CST12 ;                                         00075700
      CST31 := 2.0 * CST12 ;                                         00075800
      ETA := 0.0 ;                                                   00075900
      FOR I := 1 STEP 1 UNTIL NXENDM/2 DO                            00076000
           BEGIN                                                     00076100
           S[I] := DSIN(PI*ETA) ;                                    00076200
           C[I] := DCOS(PI*ETA) ;                                    00076300
           S[NXENDP-I] := S[I] ;                                     00076400
           C[NXENDP-I] := -C[I] ;                                    00076500
           ETA := ETA + DELX ;                                       00076600
           END ;                                                     00076700
      S[NXENDP/2] := 1.0 ;                                           00076800
      C[NXENDP/2] := 0.0 ;                                           00076900
      TSI := 0.0 ;                                                   00077000
      DTSI := DELY ;                                                 00077100
      CC := INP ;                                                    00077200
L1:   CASE CC OF BEGIN NETA;NETB;NETC;GO TO L2;NETG;GO TO L2;END;    00077300
      CC := CC + 1 ;                                                 00077400
      IF NYFILS = 2 THEN NYFILS := 1 ;                               00077500
      FOR J := NYFILS STEP  1 UNTIL NYFILF DO                        00077600
           BEGIN                                                     00077700
```

```
PITSI := PI * TSI ;                                                    00077800
VAR1 := DEXP ( PITSI ) ;                                               00077900
VAR2 := 1.0 / VAR1 ;                                                   00078000
ALF := CST25 * (VAR1 - 1.0) ;                                          00078100
IF TSI LSS 1.0@@-15 THEN                                               00078200
     BEGIN                                                             00078300
     ALF := CST25 * ( PITSI *                                          00078400
          ( 1.0 + 0.5 * PITSI * ( 1.0                                  00078500
          + CST8 * PITSI ))) ;                                         00078600
     END ;                                                             00078700
VAR3 := DEXP(-ALF * ALF ) ;                                            00078800
VAR4 := ERF( ALF ) ;                                                   00078900
VAR5 := 1.0 - VAR4 ;                                                   00079000
VAR6 := ERF( CST6 * ALF ) ;                                            00079100
VAR7 := 1.0 - VAR6 ;                                                   00079200
VAR11 := VAR2 * VAR2 ;                                                 00079300
VAR12 := CST29 * VAR2 ;                                                00079400
VAR13 := 4.0 * CST28 * VAR11 ;                                         00079500
VAR14 := VAR1 - VAR2 ;                                                 00079600
VAR17 := -VAR2 + VAR3 ;                                                00079700
VAR18 := VAR3 * ( 11.0 * VAR5 - 9.0 ) + 4.0 * VAR5                     00079800
          - 6.0 ;                                                      00079900
VAR19 := VAR3 - 1.0 ;                                                  00080000
VAR21 := 1.0 + VAR11 - 2.0 * VAR5 ;                                    00080100
VAR22 := VAR11 - VAR3 ;                                                00080200
VAR23 := VAR5 - VAR3 ;                                                 00080300
IF TSI LSS 1.0@@-15 THEN                                               00080400
     BEGIN                                                             00080500
     VAR14 := PITSI * ( 2.0 + CST8 * PITSI * PITSI ) ;                 00080600
     VAR15 := ALF * ALF ;                                              00080700
     VAR16 := VAR15 * ( - 1.0 + 0.5 * VAR15 *                          00080800
          (1.0 - CST8 * VAR15 )) ;                                     00080900
     VAR20 := 2.0 * PITSI ;                                            00081000
     VAR24 := VAR20 * ( -1.0 + 0.5 * VAR20 *                           00081100
          ( 1.0 - CST8 * CST20 )) ;                                    00081200
     VAR21 := VAR24 + 2.0 * VAR4 ;                                     00081300
     IF ALF LSS 1.0@@-7 THEN                                           00081400
          BEGIN                                                        00081500
          VAR17 := PITSI * ( 1.0 - 0.5 * PITSI *                       00081600
               (1.0 + CST8 * PITSI )) + VAR16 ;                        00081700
          VAR22 := VAR21 - VAR16 ;                                     00081800
          END ;                                                        00081900
     END ;                                                             00082000
IF ALF LSS 1.0@@-7 THEN                                                00082100
     BEGIN                                                             00082200
     VAR15 := ALF * ALF ;                                              00082300
     VAR16 := VAR15 * ( - 1.0 + 0.5 * VAR15 *                          00082400
          (1.0 - CST8 * VAR15 )) ;                                     00082500
     VAR18 := 2.0 * VAR16 - VAR4 * ( 15.0 - 11.0 *                     00082600
          VAR16 ) ;                                                    00082700
     VAR19 := VAR16 ;                                                  00082800
     VAR23 := - VAR4 - VAR16 ;                                         00082900
     END ;                                                             00083000
LEVEL :=   VAR14 + CST23 *  CST9 *                                     00083100
          ( VAR17 - CST1 * ALF * VAR5                                  00083200
          + CST23 * ( -CST10 * VAR4 +                                  00083300
          CST11 * ALF * ALF * VAR5 - 1.5 * ALF * VAR3 )) ;             00083400
INT := ALF * ( CST12 * VAR3 * VAR3 + VAR5 * ( 0.5 *                    00083500
          VAR4 - CST31) + ALF * ( VAR3 * (-CST14 * VAR5 +              00083600
          CST17) + ALF * VAR5 * (CST8 * VAR5 - CST16) ) )              00083700
```

```
                    +CST2 * ( CST26 * VAR6 + CST15 *                    00083800
                    VAR19 + 0.5 * CST8 * VAR18 ) ;                      00083900
          VAR8 := 8.0 * CST23 * T * INT ;                               00084000
          VAR9 := VAR5 * ( 3.0 + VAR2 * ( 6.0 * CST23 * ALF             00084100
                    -2.0 + VAR2 * CST23 * ( ALF * ( 4.0 -               00084200
                    6.0 * CST23 * ALF ) - CST23 ))) + VAR3 *            00084300
                    2.0 * CST2 * ( -ALF + CST24 - 2.0 *                 00084400
                    CST23 * VAR2 * ( 1.0 + VAR2 * ( 1.0 -               00084500
                    1.5 * CST23 * ALF ))) + CST22 * VAR2 *              00084600
                    VAR2 ;                                              00084700
          VAR10 := 4.0 * T * CST25 * ( VAR5 * ( VAR12 *                 00084800
                    CST13 - VAR13 * CST14 + ALF * ( -6.0 *              00084900
                    CST16 - VAR13 * CST13 - VAR3 * VAR12 *              00085000
                    3.0 * CST2 + ALF * ( -3.0 * VAR12 *                 00085100
                    CST16 + VAR3 * ( 2.0 * CST2 + VAR13 *               00085200
                    CST14 ) + ALF * ( VAR13 * CST16 -                   00085300
                    VAR5 * VAR13 * CST8 ))) + VAR5 *                    00085400
                    ( -0.5 * VAR12 +ALF * ( 2.0 + 0.5 *                 00085500
                    VAR13 + ALF * VAR12 )) - VAR3 * ( CST2 +            00085600
                    CST18 * VAR13 )) + VAR3 * ( 2.0 * CST2 *            00085700
                    ( 1.5 + CST30 ) - CST30 * VAR12 -                   00085800
                    VAR13 * CST19 + VAR3 * VAR12 * 2.0 *                00085900
                    CST3 + ALF * ( 8.0 * CST12 + CST9 *                 00086000
                    VAR12 * ( 1.0 + CST12 ) - VAR3 * ( 2.0 *            00086100
                    CST3 + CST12 * VAR13 ) - ALF * CST2 *               00086200
                    ( 2.0 + VAR13 * CST16 ))) +                         00086300
                    VAR13 * CST20 * VAR7 - VAR13 * CST21 ) ;            00086400
          FOR I := NXFILS STEP 1 UNTIL NXFILF DO                        00086500
               BEGIN                                                    00086600
               PSI[I,J] := ( LEVEL + VAR8 * C[I] ) * S[I] ;             00086700
               ZETA[I,J] := ( VAR9 + VAR10 * C[I] ) * S[I];             00086800
               FDZT[I] := PI * ( 8.0 * CST2 * CST29 +  7.0 -            00086900
                    CST9 * CST25 + REYN * ( CST2 * CST29 *              00087000
                    ( 1.0 + CST30 ) - 2.0 ) * C[I] ) *                  00087100
                    S[I] ;                                              00087200
               SLOPE[I] := -FDZT[I] * CST3 ;                            00087300
               END ;                                                    00087400
          TSI := TSI + DTSI ;                                           00087500
          END ;                                                         00087600
     GO TO L1 ;                                                         00087700
L2:  TIME1 := T ;                                                       00087800
     END ;                                                              00087900
%***********************************************************************00088000
%                                                                       00088100
  PROCEDURE INITIALIZENET ;                                             00088200
                    % INITIALIZE THE PSI NET WITH THE POTENTIAL FLOW00088300
                    %    SOLUTION, RELAX IT, COMPUTE THE SURFACE    00088400
                    %    VORTICITY,AND SET THE REST OF THE VORTICITY00088500
                    %    NET EQUAL TO ZERO                          00088600
     BEGIN                                                              00088700
     INTEGER NXP, NXM, II , I,J,K,L ;                                   00088800
     REAL   TCG, ETCG, RETCGS, FTG, DIE, SI1, PIYIN, CSIGM,             00088900
                    PIDEL,  PIE ;                                       00089000
     LABEL L1 ;                                                         00089100
     IF NITER NEQ 0 THEN BEGIN FETCHPSIZETA;GO TO L1;END;               00089200
     NXM := NXEND / 2 ;                                                 00089300
     NXP := NXM + 1 ;                                                   00089400
     SIGMA := 0.0 ;                                                     00089500
     FOR  J := 1 STEP 1 UNTIL NYENDM  DO                                00089600
          BEGIN                                                         00089700
```

```
          TCG := PI * SIGMA ;                                                    00089800
          ETCG := EXP(TCG) ;                                                     00089900
          RETCGS := 1.0 / (ETCG * ETCG) ;                                        00090000
          FTG := ETCG * (1.0 - RETCGS) ;                                         00090100
          ETA := 0.0 ;                                                           00090200
          FOR I := 1 STEP 1 UNTIL NXP  DO                                        00090300
               BEGIN                                                             00090400
               PIE := PI * ETA ;                                                 00090500
               SI1 := SIN(PIE) ;                                                 00090600
               PSI[I,J] := FTG * SI1 ;                                           00090700
               ETA := ETA + DELX ;                                               00090800
               END ;                                                             00090900
          SIGMA := SIGMA + DELY ;                                                00091000
          END;                                                                   00091100
        PIYIN := PI * YINF ;                                                     00091200
        CSIGM := EXP(PIYIN) ;                                                    00091300
        FTG := CSIGM ;                                                           00091400
        IF POTENTIAL=1 THEN                                                      00091500
          BEGIN                                                                  00091600
          RETCGS := 1.0/(CSIGM*CSIGM) ;                                          00091700
          FTG := CSIGM * (1.0 - RETCGS) ;                                        00091800
          END ;                                                                  00091900
        PIDEL := PI * DELX ;                                                     00092000
        FOR  I := 1 STEP 1 UNTIL NXP  DO                                         00092100
          BEGIN                                                                  00092200
          FI := I-1 ;                                                            00092300
          PSI[I, NYEND] :=   FTG * SIN(PIDEL * FI) ;                             00092400
          END;                                                                   00092500
        FOR J := 1 STEP 1 UNTIL NYEND   DO                                       00092600
          BEGIN                                                                  00092700
          FOR I :=  1 STEP 1 UNTIL  NXM   DO                                     00092800
               BEGIN                                                             00092900
               II := NXENDP - I ;                                                00093000
               PSI[II, J] := PSI[I,J] ;                                          00093100
               END ;                                                             00093200
          END ;                                                                  00093300
        FOR J := 1 STEP 1 UNTIL NYEND   DO                                       00093400
          BEGIN                                                                  00093500
          FOR I := 1 STEP 1 UNTIL NXEND  DO                                      00093600
               BEGIN                                                             00093700
               ZETA[I,J] := 0.0 ;                                                00093800
               END ;                                                             00093900
          END ;                                                                  00094000
        IF PP=1 THEN PRINT1 ( PSI , IPRNT ) ;                                    00094100
        IF STARTTIME EQL 0.0 THEN BEGIN STREAM; SURFACEVORTICITY; END ;          00094200
 L1:    IF RECTANGLE NEQ 1 THEN INP := 0 ;                                       00094300
        IF STARTTIME GTR 1.0@@-30 AND                                            00094400
          SECTION = 1 THEN SHORTTIMESOLUTION(STARTTIME) ;                        00094500
        END ;                                                                    00094600
%************************************************************************      00094700
%                                                                                00094800
   PROCEDURE MATINV (A,N,B,M,DETERM ) ;                                          00094900
        DOUBLE ARRAY A[*,*] , B[*,*]  ;                                          00095000
        INTEGER N,M ;                                                            00095100
        DOUBLE DETERM  ;                                                         00095200
        BEGIN                                                                    00095300
        DEFINE   IROW  = JROW #, ICOLUM  = JCOLUM  #, AMAX = T #,                00095400
                       SWAP = T # ;                                              00095500
        DOUBLE ARRAY  PIVOT[0:5] ;                                               00095600
        DOUBLE  T  ;                                                             00095700
```

```
      INTEGER  ARRAY IPIVOT[0:5],INDEX[0:5,0:2]  ;                          00095800
      INTEGER JROW, JCOLUM, L1 ,I,J,K,L ;                                   00095900
      LABEL HOP720 , HOP777 ;                                               00096000
      DETERM := 1.0 ;                                                       00096100
      FOR J := 1 STEP 1 UNTIL N  DO                                         00096200
           BEGIN                                                            00096300
           IPIVOT[J] := 0 ;                                                 00096400
           END ;                                                            00096500
      FOR I := 1 STEP 1 UNTIL N  DO                                         00096600
           BEGIN                                                            00096700
           LABEL HOP260, HOP380, HOP550 ;                                   00096800
           AMAX := 0.0 ;                                                    00096900
           FOR J := 1 STEP 1 UNTIL N  DO                                    00097000
                BEGIN                                                       00097100
                LABEL HOP105  ;                                             00097200
                IF (IPIVOT[J]-1) EQL 0 THEN GO TO HOP105 ;                  00097300
                FOR  K := 1 STEP 1 UNTIL N  DO                              00097400
                     BEGIN                                                  00097500
                     LABEL HOP100 ;                                         00097600
                     IF (IPIVOT[K] -1) GTR 0 THEN GO TO HOP777 ;            00097700
                     IF (IPIVOT[K] -1) EQL 0 THEN GO TO HOP100 ;            00097800
                     IF (DABS(AMAX) - DABS(A[J,K] )) GTR 0.0 THEN           00097900
                                    GO TO HOP100 ;                          00098000
                     IROW := J ;                                            00098100
                     ICOLUM := K ;                                          00098200
                     AMAX := A[J,K] ;                                       00098300
HOP100:              END ;                                                  00098400
HOP105:         END ;                                                       00098500
           IPIVOT[ICOLUM] := IPIVOT[ICOLUM] + 1 ;                           00098600
           IF (IROW - ICOLUM) EQL  0  THEN GO TO  HOP260 ;                  00098700
           DETERM := - DETERM ;                                             00098800
           FOR  L := 1 STEP 1 UNTIL N  DO                                   00098900
                BEGIN                                                       00099000
                SWAP := A[IROW,L] ;                                         00099100
                A[IROW,L] := A[ICOLUM,L] ;                                  00099200
                A[ICOLUM,L] := SWAP ;                                       00099300
                END ;                                                       00099400
           IF  M LEQ 0  THEN  GO TO  HOP260 ;                               00099500
           FOR L := 1 STEP 1 UNTIL M  DO                                    00099600
                BEGIN                                                       00099700
                SWAP := B[IROW,L ]  ;                                       00099800
                B[IROW,L] := B[ICOLUM,L] ;                                  00099900
                B[ICOLUM,L] := SWAP ;                                       00100000
                END;                                                        00100100
HOP260:    INDEX[I,1] := IROW ;                                             00100200
           INDEX[I,2] := ICOLUM ;                                           00100300
           PIVOT[I]  := A[ICOLUM,ICOLUM] ;                                  00100400
           DETERM := DETERM * PIVOT[I] ;                                    00100500
           IF PIVOT[I]  EQL 0  THEN GO TO  HOP720 ;                         00100600
           A[ICOLUM,ICOLUM] := 1.0 ;                                        00100700
           FOR  L := 1 STEP 1 UNTIL N  DO                                   00100800
                BEGIN                                                       00100900
                A[ICOLUM,L] := A[ICOLUM,L] / PIVOT[I]  ;                    00101000
                END ;                                                       00101100
           IF  M LEQ 0  THEN  GO TO HOP380  ;                               00101200
           FOR  L := 1 STEP 1 UNTIL M  DO                                   00101300
                BEGIN                                                       00101400
                B[ICOLUM,L] := B[ICOLUM,L] / PIVOT[I] ;                     00101500
                END ;                                                       00101600
HOP380:    FOR  L1 := 1 STEP 1 UNTIL  N   DO                                00101700
```

```
                BEGIN                                                   00101800
                IF (L1-ICOLUM)  EQL 0  THEN  GO TO HOP550 ;             00101900
                T := A[L1,ICOLUM]  ;                                    00102000
                A[L1,ICOLUM] := 0.0 ;                                   00102100
                FOR  L := 1 STEP 1 UNTIL N  DO                          00102200
                     BEGIN                                              00102300
                     A[L1,L] := A[L1,L] - A[ICOLUM,L] * T ;             00102400
                     END ;                                              00102500
                IF  M LEQ 0  THEN  GO TO HOP550 ;                       00102600
                FOR  L :=  1 STEP 1 UNTIL  M  DO                        00102700
                     BEGIN                                              00102800
                     B[L1,L] := B[L1,L] - B[ICOLUM,L] *T ;              00102900
                     END ;                                              00103000
                END;                                                    00103100
HOP550:      END  ;                                                     00103200
       FOR  I := 1 STEP 1 UNTIL N  DO                                   00103300
            BEGIN                                                       00103400
            LABEL HOP710 ;                                              00103500
            L := N+ 1 - I ;                                             00103600
            IF (INDEX[L,1] - INDEX[L,2]) EQL 0 THEN GO TO HOP710;       00103700
            JROW := INDEX[L,1] ;                                        00103800
            JCOLUM := INDEX[L,2] ;                                      00103900
            FOR K := 1 STEP 1 UNTIL  N  DO                              00104000
                 BEGIN                                                  00104100
                 SWAP := A[K,JROW] ;                                    00104200
                 A[K,JROW] := A[K,JCOLUM] ;                             00104300
                 A[K,JCOLUM] := SWAP ;                                  00104400
                 END ;                                                  00104500
HOP710:      GO TO HOP777;                                              00104600
            END;                                                        00104700
HOP720:     BEGIN                                                       00104800
            FORMAT  FMT15 ("MATRIX IS SINGULAR" )    ;                  00104900
            WRITE (LINE, FMT15 )  ;                                     00105000
            GO TO HOPEND ;                                              00105100
            END ;                                                       00105200
HOP777:     BEGIN                                                       00105300
            END ;                                                       00105400
       END ;                                                            00105500
%***********************************************************************00105600
%                                                                       00105700
  PROCEDURE  LSQ1 ( X,Y,W,N,L,A,B,M)  ;                                 00105800
       DOUBLE ARRAY  A[*,*] , B[*,*] ;                                  00105900
       INTEGER N,M,L ;                                                  00106000
       ARRAY  X[*], Y[*,*], W[*]   ;                                    00106100
       BEGIN                                                            00106200
       DOUBLE ARRAY C[0:5, 0:5]  ;                                      00106300
       DOUBLE DETERM ;                                                  00106400
       INTEGER I, J, K     ;                                            00106500
       FOR I := 1 STEP 1 UNTIL N   DO                                   00106600
            BEGIN                                                       00106700
            C[I,1] := 1.0 ;                                             00106800
            END ;                                                       00106900
       FOR J :=  2 STEP 1 UNTIL M  DO                                   00107000
            BEGIN                                                       00107100
            FOR I := 1 STEP 1 UNTIL N  DO                               00107200
                 BEGIN                                                  00107300
                 C[I,J] := C[I,J-1] * X[I] ;                            00107400
                 END ;                                                  00107500
            END;                                                        00107600
       FOR I := 1 STEP 1 UNTIL M  DO                                    00107700
```

```
            BEGIN                                                          00107800
            FOR  J := 1 STEP 1 UNTIL M DO                                  00107900
                 BEGIN                                                     00108000
                 A[I,J] :=  0.0 ;                                          00108100
                 FOR K := 1 STEP 1 UNTIL  N  DO                            00108200
                      BEGIN                                                00108300
                      A[I,J] := A[I,J] + C[K,I]*C[K,J] * W[K] ;            00108400
                      END ;                                                00108500
                 END ;                                                     00108600
            END ;                                                          00108700
       FOR  I := 1 STEP 1 UNTIL M  DO                                      00108800
            BEGIN                                                          00108900
            B[I,1] := 0.0 ;                                                00109000
            FOR K := 1 STEP 1 UNTIL  N  DO                                 00109100
                 BEGIN                                                     00109200
                 B[I,1] := B[I,1] + C[K,I] * Y[K,1]*W[K] ;                 00109300
                 END ;                                                     00109400
            END ;                                                          00109500
       MATINV ( A,M,B,L, DETERM ) ;                                        00109600
       END ;                                                               00109700
%**************************************************************************00109800
%                                                                          00109900
   PROCEDURE PRESS          ;                                              00110000
       BEGIN                                                              00110100
       REAL ARRAY RAD[0:5], W[0:5], AZETA[0:5,0:1], ZETAS[0:XD],          00110200
                  PRES[0:XD]  ;                                           00110300
       REAL PISIG, REDEL, THETA, PAREA, CD  , SAREA ;                     00110400
       INTEGER I, J, K ,L  ;                                              00110500
       LABEL L1 ;                                                         00110600
       FORMAT FMT12 ( X50,"PRESSURE DISTRIBUTION"// X45,"TIME =",         00110700
                  F7.3, X6, "NITER =", I5 // X50, "ANGLE", X9,            00110800
                  "PRESSURE"/ X50, "******", X9, "*********" //           00110900
                  ( X50, F5.1, X10, F7.3) ) ; % 104                       00111000
       FORMAT FMT13 ( //// X56,"CDP =", F9.3// X56,"CDF =",               00111100
                  F9.3 // X56,"CD  =",F9.3) ; % 109                       00111200
       FORMAT FMT72 ( X50, F5.1, X10, F7.3 ) ;                            00111300
       IF STARTTIME GTR 0.0 AND STARTTIME EQL TIME1 THEN GO TO L1;        00111400
       SIGMA := 0.0 ;                                                     00111500
       FOR  J := 1 STEP 1 UNTIL 5  DO                                     00111600
            BEGIN                                                         00111700
            PISIG := PI * SIGMA ;                                         00111800
            W[J] := 1.0 ;                                                 00111900
            RAD[J] := EXP(PISIG) ;                                        00112000
            SIGMA := SIGMA + DELY ;                                       00112100
            END ;                                                         00112200
       FOR I := 1 STEP 1 UNTIL NXEND  DO                                  00112300
            BEGIN                                                         00112400
            FOR  J := 1 STEP 1 UNTIL  5  DO                               00112500
                 BEGIN                                                    00112600
                 AZETA[J,1]:= ZETA[I,J] ;                                 00112700
                 END ;                                                    00112800
            LSQ1 ( RAD,AZETA,W,4,1,A,B,3 ) ;                              00112900
            SLOPE[I] := B[2,1] + 2.0 * B[3,1] ;                           00113000
            END;                                                          00113100
  L1:   PRES[1] := 0.0 ;                                                  00113200
       REDEL := (4.0/REYN) * (PI* DELX/3.0) ;                             00113300
       FOR  I := 3 STEP 2 UNTIL NXEND DO                                  00113400
            BEGIN                                                         00113500
            PRES[I] := PRES[I-2] - REDEL * (SLOPE[I-2] + 4.0 *            00113600
                       SLOPE[I-1] + SLOPE[I] ) ;                          00113700
```

```
          END ;                                                          00113800
     FOR  I := 1 STEP 2 UNTIL NXEND DO                                   00113900
          BEGIN                                                          00114000
          FI := I-1 ;                                                    00114100
          THETA := FI * PI * DELX ;                                      00114200
          ZETAS[I] := -PRES[I] * COS(THETA) ;                            00114300
          END ;                                                          00114400
     PAREA := 0.0 ;                                                      00114500
     L := 5 ;                                                            00114600
     IF NYENDM MOD 4 NEQ 0 THEN                                          00114700
          BEGIN                                                          00114800
          PAREA := 1.50 * (ZETAS[1] + ZETAS[3]) ;                        00114900
          L := 7 ;                                                       00115000
          END ;                                                          00115100
     FOR  I := L STEP 4 UNTIL NXEND DO                                   00115200
          BEGIN                                                          00115300
          PAREA := PAREA + (ZETAS[I-4] + 4.0 * ZETAS[I-2]                00115400
                    +ZETAS[I] ) ;                                        00115500
          END ;                                                          00115600
     PAREA := (PI * 2.0 * DELX/3.0 ) * PAREA ;                           00115700
     ZETAS[1] := 0.0 ;                                                   00115800
     FOR  I := 2 STEP 1 UNTIL NXEND  DO                                  00115900
          BEGIN                                                          00116000
          FI := I-1 ;                                                    00116100
          THETA:= FI * PI * DELX ;                                       00116200
          ZETAS[I] := ZETA[I,1] * SIN(THETA) ;                           00116300
          END ;                                                          00116400
     SAREA := 0.0 ;                                                      00116500
     FOR  I := 3 STEP 2 UNTIL NXEND  DO                                  00116600
          BEGIN                                                          00116700
          SAREA := SAREA + ( ZETAS[I-2] + 4.0 * ZETAS[I-1]               00116800
                    + ZETAS[I] ) ;                                       00116900
          END ;                                                          00117000
     SAREA := -REDEL * SAREA ;                                           00117100
     CD := PAREA + SAREA ;                                               00117200
     WRITE ( LINE [SKIP 1 ] ) ;                                          00117300
     PGNUM := PGNUM + 1 ;                                                00117400
     WRITE (LINE, FMT75, PGNUM ) ;                                       00117500
     WRITE (LINE, 16 , TITLEARRAY[*] ) ;                                 00117600
     WRITE (LINE, FMT90 ) ;                                              00117700
     WRITE (LINE, FMT12, TIME1,NITER,  FOR I := 1 STEP 1                 00117800
                    UNTIL NPRES DO [ ANGLE[I], PRES[2*I-1] ] ) ;         00117900
     IF EVEN = 1 THEN WRITE(LINE, FMT72, ANGLE[NPRES + 1] ,              00118000
                    PRES[2*NPRES] ) ;                                    00118100
     WRITE (LINE, FMT13, PAREA, SAREA, CD );                             00118200
     NPP := 0 ;                                                          00118300
     END ;                                                               00118400
%********************************************************************** 00118500
%                                                                       00118600
   PROCEDURE PRINTPARAMETERS ;                                          00118700
     BEGIN                                                              00118800
     PGNUM := PGNUM + 1 ;                                               00118900
     WRITE (LINE, FMT75, PGNUM ) ;                                      00119000
     WRITE (LINE, 16 , TITLEARRAY[*] ) ;                                00119100
     WRITE (LINE, FMT16 )  ;                                            00119200
     WRITE (LINE, FMT4) ;                                               00119300
                    % REYN::= REYNOLDS NUMBER                           00119400
                    % YINF::= LARGEST VALUE OF TS1,THE RADIAL CO-       00119500
                    %       ORDINATE                                    00119600
                    % LNP::= THE NUMBER OF TIME STEPS BETWEEN PRINT     00119700
```

```
                        %       OUTS MINUS ONE.                                  00119800
                        % NITER::= TIME STEP COUNTER, INITIALLY ZERO              00119900
                        % LNITER::= TOTAL NUMBER OF TIME STEPS FOR THIS           00120000
                        %       RUN                                              00120100
                        % TIME2::= MAXIMUM PROCESSOR TIME IN SECONDS              00120200
                        % DTIME ::= DIMENSIONLESS TIME INCREMENT                  00120300
                        % DPSI  ::= STREAM FUNCTION CONVERGENCE LIMIT             00120400
                        % NXEND ::= NUMBER OF MESH POINTS IN THE                  00120500
                        %       CIRCUMFERENTIAL  DIRECTION                       00120600
                        % NYEND ::= NUMBER OF MESH POINTS IN THE                  00120700
                        %       RADIAL DIRECTION                                 00120800
                        % WFP ::= RELAXATION PARAMETER                            00120900
                        % TIME1 ::= INITIAL DIMENSIONLESS TIME                    00121000
                        % GOP ::= A PROGRAM CONTROL VARIABLE                      00121100
                        % POTENTIAL ::= IF 1, OUTER BC TAKEN AS                   00121200
                        %       POTENTIAL FLOW, ELSE PARALLEL FLOW               00121300
        WRITE (LINE, FMT5, REYN, YINF, LNP, LNPP, LNITER, TIME2,                 00121400
                           DTIME, DPSI, TIME1, GOP, NITER,                       00121500
                           POTENTIAL, WFP, NXEND, NYEND,                         00121600
                           DELX, DELY,STARTTIME,SAVE,MODS,                       00121700
                           RECTANGLE,NXLIM1,NXLIM2,NXLIM3,                       00121800
                           NYLIM1,NYLIM2,NYLIM3,                                 00121900
                           DSV, PPZ, IREC,                                       00122000
                           SECTION ) ;                                           00122100
        END ;                                                                    00122200
%*****************************************************************************00122300
%                                                                                00122400
   PROCEDURE OUTPUTRESULTS ;                                                     00122500
        BEGIN                                                                    00122600
        IF PZ=1 THEN PRINT1 ( ZETA, ZPRNT ) ;                                    00122700
        IF PP=1 THEN PRINT1 ( PSI, PPRNT ) ;                                     00122800
        IF PPZ NEQ 1 THEN WRITE(LINE, FMT9, NIS ) ;                              00122900
        IF DSV EQL 1 THEN SAVEPSIZETA;                                           00123000
        NP := 0 ;                                                                00123100
        END ;                                                                    00123200
%*****************************************************************************00123300
%                                                                                00123400
   PROCEDURE FINALPRINTROUTINE ;                                                 00123500
        BEGIN                                                                    00123600
        IF NP NEQ 0 THEN BEGIN OUTPUTRESULTS; PRESS END ;                        00123700
        IF SAVE = 1 THEN BEGIN STOREPSIZETA;STORECONSTANTS;END;                  00123800
        LOCK(SAVEP) ;                                                            00123900
        LOCK(SAVEZ) ;                                                            00124000
        END ;                                                                    00124100
%*****************************************************************************00124200
%                                                                                00124300
   PROCEDURE PRINTPSIZETA ;                                                      00124400
        BEGIN                                                                    00124500
        LABEL HOP165, HOP190 ;                                                   00124600
        SETUPGET ;                                                               00124700
        GETPSIZETA ;                                                             00124800
        OUTPUTRESULTS ;                                                          00124900
        PRESS ;                                                                  00125000
 HOP165:NITER := NITER + 1 ;                                                     00125100
        NP := NP + 1 ;                                                           00125200
        NPP := NPP + 1 ;                                                         00125300
        TIME1 := TIME1 + DTIME ;                                                 00125400
        GETPSIZETA ;                                                             00125500
              % PRINT CURRENT VORTICITY AND STREAMFUNCTION VALUES                 00125600
              %    AND COMPUTE THE PRESSURE DISTRIBUTION                         00125700
```

```
                    %    WHENEVER REQUESTED BY LNP                    00125800
      IF NP EQL LNP THEN OUTPUTRESULTS;                                00125900
      IF NPP EQL LNPP THEN PRESS ;                                     00126000
                    % IF PROGRAM RUNS OVERTIME GO TO END OF JOB       00126100
      IF  GOP GEQ 1.5  THEN  GO TO HOP190 ;                            00126200
                    % IF DESIRED NUMBER OF TIME STEPS HAVE            00126300
                    %    BEEN COMPLETED THEN GO TO END OF JOB        00126400
      IF NITER GEQ LNITER  THEN  GO TO HOP190 ;                        00126500
                                                                       00126600
                    % OTHERWISE COMPUTE THE NEXT TIME STEP            00126700
      GO TO HOP165 ;                                                   00126800
HOP190:END ;                                                           00126900
%**********************************************************************00127000
%                                                                      00127100
   PROCEDURE MAIN ;                                                    00127200
      BEGIN                                                            00127300
      LABEL  HOP165, HOP190, HOP150 ;                                  00127400
      IF PPZ EQL 1 AND NITER EQL 0 THEN GO TO HOP150 ;                 00127500
      INITIALIZEPROGRAM ;                                              00127600
HOP150:COMPUTECONSTANTS ;                                              00127700
      PRINTPARAMETERS ;                                                00127800
      IF PPZ EQL 1 THEN BEGIN PRINTPSIZETA;GO TO HOP190;END;           00127900
      INITIALIZENET ;                                                  00128000
      IF SECTION NEQ 1  THEN GO TO HOP165 ;                            00128100
      OUTPUTRESULTS ;                                                  00128200
      PRESS ;                                                          00128300
                    % MAIN PROGRAM LOOP                               00128400
HOP165:NITER := NITER + 1 ;                                            00128500
      NP := NP + 1 ;                                                   00128600
      NPP := NPP + 1 ;                                                 00128700
      TIME1 := TIME1 + DTIME ;                                         00128800
                    % COMPUTE THE NEW VORTICITY BY THE                00128900
                    %    ALTERNATING DIRECTION IMPLICIT METHOD       00129000
      ADI ;                                                            00129100
                    % RELAX THE STREAM FUNCTION WITH THE              00129200
                    %    ACCELERATED LIEBMAN METHOD                  00129300
      STREAM ;                                                         00129400
                    % COMPUTE THE SURFACE VORTICITY                   00129500
      SURFACEVORTICITY ;                                               00129600
            % PRINT CURRENT VORTICITY AND STREAMFUNCTION VALUES        00129700
            %    AND COMPUTE THE PRESSURE DISTRIBUTION               00129800
                    %    WHENEVER REQUESTED BY LNP                   00129900
      IF NP EQL LNP THEN OUTPUTRESULTS;                                00130000
      IF NPP EQL LNPP THEN PRESS ;                                     00130100
                    % IF PROGRAM RUNS OVERTIME GO TO END OF JOB       00130200
      IF  GOP GEQ 1.5  THEN  GO TO HOP190 ;                            00130300
                    % IF DESIRED NUMBER OF TIME STEPS HAVE            00130400
                    %    BEEN COMPLETED THEN GO TO END OF JOB        00130500
      IF NITER GEQ LNITER  THEN  GO TO HOP190 ;                        00130600
                                                                       00130700
                    % OTHERWISE COMPUTE THE NEXT TIME STEP            00130800
      GO TO HOP165 ;                                                   00130900
HOP190:END ;                                                           00131000
%**********************************************************************00131100
%**********************************************************************00131200
%                                                                      00131300
      MAIN ;                                                           00131400
HOP999:FINALPRINTROUTINE ;                                             00131500
HOPEND:TYMOUT ;                                                        00131600
      END ;                                                            00131700
```

```
%*************************************************************************************00131800
%                                                                                     00131900
   END.                                                                               00132000
```

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
 40.0,/    REYNOLDS NUMBER                                                           00000100
 1.5,/     MAXIMUM VALUE OF THE RADIAL COORDINATE                                    00000200
  1,/      PSI-ZETA PRINTOUT CONTROL VARIABLE                                        00000300
 1,/       LNPP CONTROLS OUTPUT OF PRESSURE DIST.                                    00000400
   3,/     MAXIMUM NUMBER OF ITERATIONS THIS RUN                                     00000500
 6000.0,/  LIMIT ON MAXIMUM RUN TIME                                                 00000600
 0.04,/          DEMENTIONLESS TIME INCREMENT                                        00000700
 0.00001,/       ITERATION ERROR LIMIT                                               00000800
 0.0,/           INITIAL DIMETIONLESS TIME                                           00000900
 1.0,/           PROGRAM CONTROL VARIABLE ?                                          00001000
 0,/             INITIAL INTERATION NUMBER                                           00001100
 0,/       SELECTS THE OUTER BOUNDARY CONDITION                                      00001200
 0.00,/    START TIME OF SHORT TIME SOLUTION                                         00001300
 0,/       IF SAVE=1 A COPY OF PSI AND ZETA SAVED                                    00001400
 1,/       IF MODS=1 MODIFIED STREAM USED                                            00001500
 0,/       IF DSV=1 THEN DISK FILE OF PSI-ZETA SAVED                                 00001600
 0,/       IF PPZ=1 THEN DISK FILE OF P-Z PRINTED OUT                                00001700
 1, 1, /  PZ,PP IF=1 PRINT ZETA,PSI WHEN OUTPUT                                     00001800
  9,/   NUMBER OF MESH POINTS IN CIRCUM. DIRECTION                                   00001900
 17,/   NUMBER OF MESH POINTS IN RADIAL DIRECTION                                   00002000
 1,/             RECTANGLE IF EQUAL 1 OTHERWISE MESH                                00002100
  2, 3, 4, /       X LIMITS                                                         00002200
  48, 49, 50, /    Y LIMITS                                                         00002300
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

# APPENDIX D

```
                    % CICHY/DIRECT                                          00000100
BEGIN                                                                       00000200
REAL  DPSI, TIME2, GOP, YINF, TIME1, REYN, DTIME,STARTTIME ;                00000300
INTEGER NXEND,NYEND,XD,YD,ZD,NITER,LNP,LNITER,POTENTIAL,                    00000400
                    SAVE,LNPP,RECTANGLE,NXLIM1,NXLIM2,                      00000500
                    NXLIM3,NYLIM1,NYLIM2,NYLIM3,MODS,                       00000600
                    IQX,IQY,PPZ,DSV,PZ,PP ;                                 00000700
FILE DATA(MAXRECSIZE=14,BLOCKSIZE=420,KIND=1,ACCESS=0,                      00000800
                    TITLE="CICHY/DATAMASTER.") ;                            00000900
READ(DATA,/,REYN,YINF,LNP,LNPP,LNITER,TIME2,DTIME,DPSI,TIME1,               00001000
                    GOP,NITER,POTENTIAL,STARTTIME,SAVE,MODS,                00001100
                    DSV,PPZ,PZ,PP,NXEND,NYEND,RECTANGLE,NXLIM1,             00001200
                    NXLIM2,NXLIM3,NYLIM1,NYLIM2,NYLIM3 ) ;                  00001300
IQX := LN(NXEND-1) / LN(2) ;                                                00001400
IQY := LN(NYEND-1) / LN(2) ;                                                00001500
NXEND := 2 ** IQX + 1 ;                                                     00001600
NYEND := 2 ** IQY + 1 ;                                                     00001700
XD := NXEND ;                                                               00001800
YD := NYEND ;                                                               00001900
ZD := NYEND ;                                                               00002000
IF XD GTR YD THEN ZD := NXEND ;                                             00002100
     BEGIN                                                                  00002200
     DOUBLE ARRAY  ZETA[0:XD,0:YD] , PSI[0:XD,0:YD] ,                       00002300
                    EE[0:YD], ES[0:YD], RE[0:YD] ,RS[0:YD],                 00002400
                    RMOS[0:YD] , SMDT[0:YD] , F[0:ZD],                      00002500
                    D[0:ZD] , G[0:ZD], A[0:5,0:5], B[0:5,0:1],              00002600
                    SI[0:513], Z[0:1025], Y[0:1025], AKX[0:1025],           00002700
                    KORE[0:XD,0:YD], SLOPE[0:XD] ;                          00002800
     REAL ARRAY  ANGLE[0:XD]  ;                                             00002900
     INTEGER ARRAY  INDEX[0:513] ;                                          00003000
     ARRAY TITLEARRAY[0:17] ;                                               00003100
     DOUBLE  DP1, DP4, RXS, RYS, RDS4, RERX2, RERY2, WFBZ2,                 00003200
                    DELX, DELY, PI, DELXS, DELYS, PIS, RERX1,               00003300
                    RERX4, RERY1, RERY4, SIGMA, DELPI , RXYS,               00003400
                    HX, HY, HXY2, POTFAC ;                                  00003500
     REAL ETA, DEL1, FK, DELTI, TYME, FI, WFBZ, WFP,                        00003600
                    PROTIM, IOTIM, ELAPTIM, TOTTIM ;                        00003700
     INTEGER IT1, ONE, NP, NPP, K, J, ITZ, PROTIMIN, IOTIMIN,               00003800
                    I, L, NXENDM, NXENDP, NYENDM, NYENDP,                   00003900
                    ELAPTIMIN, TOTTIMSTART, PROTIMOUT,                      00004000
                    IOTIMOUT, ELAPTIMOUT, PGNUM, HALF,                      00004100
                    NTEST, EVEN, NPRES, NIS, TOTTIMEND,                     00004200
                    N2, N3, N4, N7, ISL, L1, IBC, NX, NY,                   00004300
                    NXENDM2, NYENDM2, SECTION, IREC ;                       00004400
     POINTER FTITLE ;                                                       00004500
     ALPHA  PPRNT, ZPRNT, EPRNT, APRNT, BPRNT, NPRNT, IPRNT;                00004600
     LABEL  HOP999, HOPEND ;                                                00004700
     FILE LINE(KIND=135,MYUSE=2,MAXRECSIZE=22,BUFFERS=2,                    00004800
                    INTMODE=3 ) ;                                           00004900
     FILE CARD(KIND=9,INTMODE=3,MAXRECSIZE=10 )  ;                          00005000
     FILE STOREP ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00005100
                    SAVEFACTOR=99,AREAS=20,AREASIZE=30,                     00005200
                    TITLE="CICHY/PSI.") ;                                   00005300
     FILE STOREZ ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00005400
                    SAVEFACTOR=99,AREAS=20,AREASIZE=30,                     00005500
                    TITLE="CICHY/ZETA.") ;                                  00005600
     FILE STOREC( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,              00005700
                    SAVEFACTOR=99,AREAS=1,AREASIZE=50,                      00005800
                    TITLE="CICHY/CONSTANTS.") ;                             00005900
     FILE SAVEP  ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00006000
```

```
                    ACCESS=1,SAVEFACTOR=99,                                  00006100
                    MAXRECSIZE=2*YD+2,TITLE="C1CHY/SPS1.") ;                 00006200
      FILE SAVEZ  ( KIND=DISK,FILETYPE=7,INTMODE=0,PROTECTION=0,             00006300
                    ACCESS=1,SAVEFACTOR=99,                                  00006400
                    MAXRECSIZE=2*YD+2,TITLE="C1CHY/SZETA.") ;                00006500
      FORMAT  FMT1 (F10.0,2110,F10.0) ;  % 102                               00006600
      FORMAT  FMT2 (6F10.0) ;            % 103                               00006700
      FORMAT  FMT3 (2F10.3, 3I10 ) ;     % 105                               00006800
      FORMAT  FMT4 (X56, "PARAMETERS"/) ; % 107                             00006900
      FORMAT  FMT5 (X49,"REYN    =",F13.7/ X49,"YINF    =",F13.7/             00007000
                    X49,"LNP     =", I5  / X49,"LNPP    =", I5  /           00007100
                    X49,"LNITER =", I5  /                                    00007200
                    X49,"TIME2  =",F13.7/ X49,"DTIME   =",F13.7//            00007300
                    X49,"TIME1  =",F13.7/                                    00007400
                    X49,"NITER  =", I5  /                                    00007500
                    X49,"POTENT =", I5  //                                   00007600
                    X49,"NXEND  =", I5  / X49,"NYEND   =", I5  /             00007700
                    X49,"IQX    =", I5  / X49,"IQY     =", I5  /             00007800
                    X49,"DELX   =",F13.7/ X49,"DELY    =",F13.7//            00007900
                    X49,"STTIME =",F13.7/ X49,"SAVE    =", I5  //            00008000
                    X49,"DSV    =", I5  / X49,"PPZ     =", I5  /             00008100
                    X49,"IREC   =", I5 // X49,"SECTION=", I5  ) ;            00008200
      FORMAT  FMT6 ("  ") ;                                                  00008300
      FORMAT  FMT7 ("TIME =", F10.2) ;   % 108                               00008400
      FORMAT  FMT8 (10F8.4) ;                                                00008500
      FORMAT  FMT16 (/////////) ;                                            00008600
      FORMAT  FMT75 ( //// X7,"C1CHY",X108, "  PAGE ", I3// ) ;              00008700
      FORMAT  FMT90 ( // ) ;                                                 00008800
%*************************************************************************00008900
%                                                                          00009000
   PROCEDURE DATIME ;                                                      00009100
      BEGIN                                                                00009200
      VALUE ARRAY A(60"   SUN", 60"   MON", 60"  TUES",60"WEDNES",         00009300
                    60" THURS", 60"   FRI", 60" SATUR") ;                  00009400
      INTEGER D,H,M,MIN,Q,P,Y , TIMEON ;                                   00009500
      LABEL OWT ;                                                          00009600
      TIMEON := TIME(1) ;                                                  00009700
      Y := TIME(0) ;                                                       00009800
      D := Y.[17:6]*100 + 10*Y.[11:6] + Y.[5:6]  ;                         00009900
      Y := Y.[29:6]*10 + Y.[23:6] ;                                        00010000
      FOR H :=31,IF Y MOD 4=0 THEN 29                                      00010100
                    ELSE 28,31,30,31,30,31,31,30,31,30 DO                  00010200
           IF D LEQ H THEN GO TO OWT                                       00010300
                    ELSE BEGIN D:=D-H;M:=M+1 END ;                         00010400
 OWT:   H:=TIMEON DIV 216000 ;                                             00010500
      MIN:=(TIMEON DIV 3600) MOD 60 ;                                      00010600
      IF M LSS 2 THEN BEGIN Q:=M+11;P:=Y-1 END                             00010700
                    ELSE BEGIN Q:=M-1;P:=Y END ;                           00010800
      M:=M+1 ;                                                             00010900
      FTITLE:=POINTER(TITLEARRAY,6) ;                                      00011000
      REPLACE FTITLE BY 6" " FOR 132  ;                                    00011100
      REPLACE FTITLE+85     BY       POINTER(A[((Q*26-2) DIV               00011200
                    10+D+P+P.[11:10]+1) MOD 7],6) FOR 6, 6"DAY. ",         00011300
                    M FOR 2 DIGITS,6"/", D FOR 2 DIGITS, 6"/",             00011400
                    Y FOR 2 DIGITS, 6", ",(12*REAL(Q:=H MOD 12=0)+Q)       00011500
                    FOR  2 DIGITS, 6":",MIN FOR 2 DIGITS,6" " ,            00011600
                    IF H GEQ 12 THEN 60"P"  ELSE 60"A" FOR 1, 6"M" ;       00011700
      END DATIME ;                                                         00011800
%*************************************************************************00011900
%                                                                          00012000
```

```
   PROCEDURE  PRINT1( NETTYPE, NETNAME ) ;                                  00012100
                    % PRINTOUT ROUTINE                                      00012200
        DOUBLE ARRAY  NETTYPE[*,*] ;                                        00012300
        ALPHA  NETNAME ;                                                    00012400
        BEGIN                                                               00012500
        ALPHA  DESIG ;                                                      00012600
        INTEGER NC, II, M1, M2 , KK , NXMARK, NXLIM, SECHAF ;               00012700
        LABEL HOP512, HOP503  , HOP504 ;                                    00012800
        FORMAT FMT10( X20,"NITER ="I5,X23,  "SECTION NUMBER  ", I3, A4,00012900
                    X5, A4, X19, "TIME =", F7.3/ ) ;                        00013000
        FORMAT FMT11( 11E12.4 ) ;                                           00013100
        NC := NYEND / 11  + 1 ;                                             00013200
        FOR  II := 1 STEP 1 UNTIL NC   DO                                   00013300
             BEGIN                                                          00013400
             SECHAF := 0 ;                                                  00013500
             M1 := 11 * II - 10 ;                                           00013600
             IF (M1 - NYEND) GTR 0.0  THEN GO TO HOP512 ;                   00013700
             M2 := M1 + 10 ;                                                00013800
             IF (M2 - NYEND) LEQ 0.0  THEN GO TO HOP504 ;                   00013900
             M2 := NYEND ;                                                  00014000
 HOP504:     IF HALF=1 THEN BEGIN NXMARK:=NXEND/2.0; NXLIM:= 1 ;            00014100
                    DESIG := APRNT: END                                     00014200
                    ELSE BEGIN NXMARK:=NXEND: NXLIM:= 1 ;                   00014300
                    DESIG := NPRNT ; END ;                                  00014400
 HOP503:     WRITE (LINE [SKIP 1 ] ) ;                                      00014500
             PGNUM := PGNUM + 1 ;                                           00014600
             WRITE (LINE, FMT75, PGNUM ) ;                                  00014700
             WRITE ( LINE, 16 , TITLEARRAY[*] ) ;                           00014800
             WRITE (LINE, FMT90 ) ;                                         00014900
             WRITE (LINE,FMT10, NITER, II, DESIG, NETNAME, TIME1 ) ;        00015000
             FOR  I := NXLIM STEP 1 UNTIL NXMARK  DO                        00015100
                  BEGIN                                                     00015200
                  WRITE(LINE, FMT11, FOR J := M1 STEP 1 UNTIL M2 DO         00015300
                       NETTYPE[I,J]) ;                                      00015400
                  END ;                                                     00015500
             IF SECHAF = 1 THEN GO TO HOP512 ;                              00015600
             IF HALF=1 THEN BEGIN SECHAF:=1; NXLIM:= NXMARK + 1;            00015700
                    NXMARK:=NXEND;  DESIG := BPRNT ;                        00015800
                    GO TO HOP503; END ;                                     00015900
 HOP512:     END ;                                                          00016000
        END ;                                                               00016100
%***********************************************************************00016200
%                                                                           00016300
   PROCEDURE TYMIN;                                                         00016400
                    % INITIALIZE THE TIMING ROUTINE                         00016500
        BEGIN                                                               00016600
        PROTIMIN:= TIME(2) ;                                                00016700
        IOTIMIN := TIME(3) ;                                                00016800
        ELAPTIMIN := TIME(1) ;                                              00016900
        END ;                                                               00017000
%***********************************************************************00017100
%                                                                           00017200
   PROCEDURE TYMOUT ;                                                       00017300
                    % OUTPUT TIMING INFORMATION                             00017400
        BEGIN                                                               00017500
        FORMAT FMT50 (X4,"****","PROCESS TIME (SEC) =", F6.2, X2, "***",00017600
                    "I/O TIME (SEC) =",F5.2, X2, "***",                     00017700
                    "ELAPSED TIME (SEC) =", F6.2, X2, "**",                 00017800
                    "TOTAL ELAPSED TIME (MINUTES) =", F5.2 ) ;              00017900
        FORMAT FMT51 (//////////);                                          00018000
```

```
        PROTIMOUT := TIME(2) ;                                          00018100
        IOTIMOUT := TIME(3) ;                                           00018200
        ELAPTIMOUT := TIME(1) ;                                         00018300
        PROTIM := (PROTIMOUT - PROTIMIN)/60 ;                           00018400
        IOTIM := (IOTIMOUT - IOTIMIN)/60 ;                              00018500
        ELAPTIM := (ELAPTIMOUT - ELAPTIMIN)/60 ;                        00018600
        TOTTIMEND := TIME(1) ;                                          00018700
        TOTTIM  := (TOTTIMEND - TOTTIMSTART)/3600 ;                     00018800
        WRITE(LINE,FMT51) ;                                             00018900
        WRITE( LINE, FMT50, PROTIM, IOTIM, ELAPTIM, TOTTIM);            00019000
        END ;                                                           00019100
%***************************************************************************00019200
%                                                                       00019300
   PROCEDURE CHECKTIMELIMIT ;                                           00019400
        BEGIN                                                           00019500
                        % ELAPSED TIME OF JOB IN 1/60 SECS              00019600
        ITZ := TIME(2) ;                                                00019700
                        % ELAPSED TIME IN SECS                          00019800
        TYME := ITZ / 60 ;                                              00019900
                        % COMPARE TO TIME LIMIT                         00020000
        IF TYME GTR TIME2 THEN GO TO HOP999 ;                           00020100
        END ;                                                           00020200
%***************************************************************************00020300
%                                                                       00020400
   PROCEDURE  SAVEPSIZETA ;                                             00020500
        BEGIN                                                           00020600
        INTEGER RECORD ;                                                00020700
        IF IREC=1 THEN                                                  00020800
             BEGIN                                                      00020900
             SAVEP.AREASIZE := XD ;                                     00021000
             SAVEP.AREAS := 100 ;                                       00021100
                                                                        00021200
             WRITE(SAVEP[0],*,REYN,YINF,                                00021300
                        DTIME,DPSI,POTENTIAL,                           00021400
                        WFP,NXEND,NYEND,DELX,DELY,STARTTIME,            00021500
                        MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,            00021600
                        NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;           00021700
             SAVEZ.AREASIZE := XD ;                                     00021800
             SAVEZ.AREAS := 100 ;                                       00021900
             WRITE(SAVEZ[0],*,REYN,YINF,                                00022000
                        DTIME,DPSI,POTENTIAL,                           00022100
                        WFP,NXEND,NYEND,DELX,DELY,STARTTIME,            00022200
                        MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,            00022300
                        NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;           00022400
             LOCK(SAVEP) ;                                              00022500
             LOCK(SAVEZ) ;                                              00022600
             SAVEP.AREASIZE := 0 ;                                      00022700
             SAVEP.AREAS := 0 ;                                         00022800
             SAVEZ.AREASIZE := 0 ;                                      00022900
             SAVEZ.AREAS := 0 ;                                         00023000
             END ;                                                      00023100
        I := 1 ;                                                        00023200
        FOR RECORD:=IREC STEP 1 UNTIL XD-1+IREC DO                      00023300
             BEGIN                                                      00023400
             WRITE(SAVEP[RECORD],2*YD+2,PSI[I,*]);                      00023500
             WRITE(SAVEZ[RECORD],2*YD+2,ZETA[I,*]);                     00023600
             I := I + 1 ;                                               00023700
             END ;                                                      00023800
        IREC := IREC + XD ;                                             00023900
        CLOSE(SAVEP) ;                                                  00024000
```

```
        CLOSE(SAVEZ) ;                                                  00024100
        END ;                                                           00024200
%***************************************************************************00024300
%                                                                       00024400
    PROCEDURE STOREPSIZETA ;                                            00024500
        BEGIN                                                           00024600
        STOREP.MAXRECSIZE:=2*YD+2 ;                                     00024700
        STOREP.AREAS:=20 ;                                              00024800
        STOREP.AREASIZE:=30 ;                                           00024900
        FOR I:=1 STEP 1 UNTIL XD DO WRITE(STOREP,2*YD+2,PSI[I,*]) ;     00025000
        LOCK(STOREP) ;                                                  00025100
        STOREZ.MAXRECSIZE:=2*YD+2 ;                                     00025200
        STOREZ.AREAS:=20 ;                                              00025300
        STOREZ.AREASIZE:=30 ;                                           00025400
        FOR I:=1 STEP 1 UNTIL XD DO WRITE(STOREZ,2*YD+2,ZETA[I,*]) ;    00025500
        LOCK(STOREZ) ;                                                  00025600
        END ;                                                           00025700
%***************************************************************************00025800
%                                                                       00025900
    PROCEDURE STORECONSTANTS ;                                          00026000
        BEGIN                                                           00026100
        STOREC.MAXRECSIZE := 50 ;                                       00026200
        STOREC.AREAS := 1 ;                                             00026300
        STOREC.AREASIZE := 50 ;                                         00026400
        WRITE(STOREC, * , REYN,YINF,DTIME,DPSI,TIME1,GOP,               00026500
                    NITER,NXEND,NYEND,SECTION,RECTANGLE,                00026600
                    NXLIM1,NXLIM2,NXLIM3,NYLIM1,NYLIM2,                 00026700
                    NYLIM3,MODS, IQX,IQY,IREC ) ;                       00026800
        LOCK(STOREC) ;                                                  00026900
        END ;                                                           00027000
%***************************************************************************00027100
%                                                                       00027200
    PROCEDURE  SETUPGET ;                                               00027300
        BEGIN                                                           00027400
        IF SECTION = 1 THEN                                             00027500
            BEGIN                                                       00027600
            READ (SAVEP[0],*,REYN,YINF,                                 00027700
                      DTIME,DPSI,POTENTIAL,                             00027800
                      WFP,NXEND,NYEND,DELX,DELY,STARTTIME,              00027900
                      MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,              00028000
                      NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;             00028100
            READ (SAVEZ[0],*,REYN,YINF,                                 00028200
                      DTIME,DPSI,POTENTIAL,                             00028300
                      WFP,NXEND,NYEND,DELX,DELY,STARTTIME,              00028400
                      MODS,RECTANGLE,NXLIM1,NXLIM2,NXLIM3,              00028500
                      NYLIM1,NYLIM2,NYLIM3,TITLEARRAY[*]) ;             00028600
            END ;                                                       00028700
        END ;                                                           00028800
%***************************************************************************00028900
%                                                                       00029000
    PROCEDURE GETPSIZETA ;                                              00029100
        BEGIN                                                           00029200
        INTEGER RECORD ;                                                00029300
        I := 1 ;                                                        00029400
        FOR RECORD := IREC STEP 1 UNTIL XD-1+IREC  DO                   00029500
            BEGIN                                                       00029600
            READ(SAVEP[RECORD],2*YD+2,PSI[I,*]);                        00029700
            READ(SAVEZ[RECORD],2*YD+2,ZETA[I,*]);                       00029800
            I := I + 1 ;                                                00029900
            END ;                                                       00030000
```

```
        IREC := IREC + XD ;
        END ;
%*************************************************************************
%
   PROCEDURE FETCHPSIZETA ;
        BEGIN
        STOREP.AREAS:=0 ;
        STOREP.AREASIZE:=0 ;
        STOREZ.AREAS:=0 ;
        STOREZ.AREASIZE:=0 ;
        IF STOREP.PRESENT THEN
             BEGIN
             LABEL EOF1,EOF2 ;
             FOR I:= 1 STEP 1 UNTIL XD DO
                       READ(STOREP,2*YD+2,PSI[I,*])[EOF1] ;
 EOF1:       FOR I:= 1 STEP 1 UNTIL XD DO
                       READ(STOREZ,2*YD+2,ZETA[I,*])[EOF2] ;
 EOF2:       I:= I-1;
             J:= (STOREP.MAXRECSIZE) DIV 2 ;
             CLOSE (STOREP) ;
             CLOSE (STOREZ) ;
             END ;
        END ;
%*************************************************************************
%
   PROCEDURE FETCHCONSTANTS ;
        BEGIN
        STOREC.AREAS := 0 ;
        STOREC.AREASIZE :=  0 ;
        IF STOREC.PRESENT THEN
             BEGIN
             READ(STOREC, * , REYN,YINF,DTIME,DPSI,TIME1,GOP,
                       NITER,NXEND,NYEND,SECTION,RECTANGLE,
                       NXLIM1,NXLIM2,NXLIM3,NYLIM1,NYLIM2,
                       NYLIM3,MODS,IQX,IQY ,IREC) ;
             CLOSE(STOREC) ;
             END ;
        END ;
%*************************************************************************
%
   PROCEDURE INITIALIZEPROGRAM ;
        BEGIN
        LABEL SKIP ;
        TOTTIMSTART := TIME(1) ;
        TYMIN ;
        DATIME ;
        SECTION := 0 ;
        NP := 0 ;
        NPP := 0 ;
        IF PPZ=1 AND NITER=0 THEN GO TO SKIP ;
        FETCHCONSTANTS ;
 SKIP:  SECTION := SECTION + 1 ;
        IF SECTION EQL 1 THEN IREC := 1 ;
        PGNUM := 0 ;
        IT1 := TIME(2) ;
        LNITER := LNITER + NITER ;
        REPLACE FTITLE+15 BY
             6"KOR/DIRECT - FLOW AROUND A CYLINDER - REVISED 05/17/72";
        END ;
%*************************************************************************
```

```
%                                                                       00036100
   PROCEDURE  SETF67 (IQ) ;                                             00036200
        INTEGER IQ ;                                                    00036300
        BEGIN                                                           00036400
        INTEGER  N5,I,K1,K,IL;                                          00036500
        LABEL  L11,L10,L9 ;                                             00036600
        N3 := 2**IQ ;                                                   00036700
        N7 := N3/2 ;                                                    00036800
        N5 := N3/4 ;                                                    00036900
        I := 1 ;                                                        00037000
        INDEX[I] := N5 ;                                                00037100
        SI[I] := 1.0 / SQRT(2.0) ;                                      00037200
        K := I ;                                                        00037300
        I := I+1 ;                                                      00037400
%                                                                       00037500
 L11:   IL := I ;                                                       00037600
        IF I EQL N7  THEN GO TO L9 ;                                    00037700
%                                                                       00037800
 L10:   K1 := INDEX[K]/2 ;                                              00037900
        INDEX[I] := K1 ;                                                00038000
        SI[I] := DSIN(PI*K1/N3) ;                                       00038100
        K1 := N7 - K1 ;                                                 00038200
        I := I+1 ;                                                      00038300
        INDEX[I] := K1 ;                                                00038400
        SI[I] := DSIN(PI*K1/N3) ;                                       00038500
        K := K+1 ;                                                      00038600
        I := I+1 ;                                                      00038700
        IF  K NEQ IL  THEN GO TO L10 ;                                  00038800
        GO TO L11 ;                                                     00038900
 L9:    END ;                                                          00039000
%***********************************************************************00039100
%                                                                       00039200
   PROCEDURE SETPT1 ;                                                   00039300
        BEGIN                                                           00039400
        INTEGER I ;                                                     00039500
        DOUBLE B, F1, F2, F3, F ;                                       00039600
        HX := DELX ;                                                    00039700
        HY := DELY ;                                                    00039800
        SETF67(IQX) ;                                                   00039900
        NX := 2**IQX ;                                                  00040000
        NY := 2**IQY ;                                                  00040100
%                                                                       00040200
        POTFAC := 2.0/NX ;                                              00040300
        HXY2 := (HX/HY)**2 ;                                            00040400
        B := 1.0/HXY2 ;                                                 00040500
        F1 := (B+1)*(3.0*B+1) ;                                         00040600
        F2 := B*B ;                                                     00040700
        F3 := 4.0*B*(B+1) ;                                             00040800
        F := PI/NX ;                                                    00040900
        FOR I:= 2   STEP 1 UNTIL NXENDM DO                              00041000
             BEGIN                                                      00041100
             AKX[I] :=2.0*(F1-F3*DCOS(F*(I-1)) + F2*DCOS(F*(I-1)*2.0)):00041200
             END ;                                                      00041300
%                                                                       00041400
%                                                                       00041500
        AKX[1] := 2.0 ;                                                 00041600
        AKX[NXEND] := 2.0*(F1+F3+F2) ;                                  00041700
        END ;                                                           00041800
%***********************************************************************00041900
%                                                                       00042000
```

```
PROCEDURE COMPUTECONSTANTS :                                                    00042100
     BEGIN                                                                      00042200
     EPRNT :=6"ERRR" ;                                                          00042300
     PPRNT :=6"PSI " ;                                                          00042400
     ZPRNT :=6"ZETA" ;                                                          00042500
     APRNT :=6"A   " ;                                                          00042600
     BPRNT :=6"B   " ;                                                          00042700
     NPRNT :=6"    " ;                                                          00042800
     IPRNT :=6"IPSI" ;                                                          00042900
                    % IF NXEND EXCEEDS 50 THE OUTPUT IS SPLIT                   00043000
                    %    AND PUT ON TWO PAGES.                                  00043100
     IF NXEND GTR 50 THEN HALF := 1 ELSE HALF := 0 ;                            00043200
     NTEST := NXEND / 2 ;                                                       00043300
     IF ( 2 * NTEST - NXEND) = 0 THEN EVEN := 1 ELSE EVEN := 0 ;                00043400
     NXENDM := NXEND - 1 ;                                                      00043500
     NYENDM := NYEND - 1 ;                                                      00043600
     NXENDP := NXEND +1 ;                                                       00043700
     NYENDP := NYEND +1 ;                                                       00043800
     NXENDM2 := NXEND - 2 ;                                                     00043900
     NYENDM2 := NYEND - 2 ;                                                     00044000
     IF EVEN = 1 THEN NPRES := NXEND/2 ELSE NPRES := NXENDM/2 + 1 ;             00044100
     DELX := 1.0 / NXENDM ;                                                     00044200
     DELY := YINF / NYENDM ;                                                    00044300
     DEL1 := 180.0 * DELX ;                                                     00044400
     ONE  := 1 ;                                                                00044500
     PI := 4 * DARCTAN(ONE)  ;                                                  00044600
     PIS := PI * PI ;                                                           00044700
     DELPI := 2.0 * PI * DELY ;                                                 00044800
     FOR  K := 1 STEP 1 UNTIL NYENDM  DO                                        00044900
          BEGIN                                                                 00045000
          FK := K - 1 ;                                                         00045100
          SIGMA := FK * DELPI ;                                                 00045200
          RMOS[K]:= PIS * DEXP(SIGMA) ;                                         00045300
          END;                                                                  00045400
     DELXS := DELX * DELX ;                                                     00045500
     DELYS := DELY * DELY ;                                                     00045600
     WFBZ2 :=  1.0 / (2.0 *  PIS * DELYS) ;                                     00045700
     RXS := 1.0 / DELXS ;                                                       00045800
     RYS := 1.0 / DELYS ;                                                       00045900
     RXYS := 1.0 / (2.0 * (RXS + RYS) ) ;                                       00046000
     RDS4 := 0.25 / (DELX * DELY) ;                                             00046100
     RERX1 := RXS / REYN ;                                                      00046200
     RERX2 := 2.0 * RERX1 ;                                                     00046300
     RERX4 := 4.0 * RERX1 ;                                                     00046400
     RERY1 := RYS / REYN ;                                                      00046500
     RERY2 := 2.0 * RERY1 ;                                                     00046600
     RERY4 := 4.0 * RERY1 ;                                                     00046700
                    % COMPUTE OVERRELAXATION PARAMETER                          00046800
          BEGIN                                                                 00046900
          REAL DR,DRE ;                                                         00047000
          DR := 0.5 * LN(0.5*((1.0/NXEND)**2 + (1.0/NYEND)**2));                00047100
          DRE := EXP(DR);                                                       00047200
          WFP := 2.0 / (1.0 + PI * DRE) ;                                       00047300
```

```
          END ;                                                          00047400
        DP1 := 1.0 - WFP ;                                               00047500
        DP4 := WFP * RXYS ;                                              00047600
        FOR  J := 1  STEP 1 UNTIL NPRES  DO                              00047700
          BEGIN                                                          00047800
          FI := 2 * J - 2 ;                                              00047900
          ANGLE[J] := DEL1 * FI ;                                        00048000
          END ;                                                          00048100
        IF EVEN = 1 THEN ANGLE[ NPRES + 1 ] := DEL1 * (FI + 1) ;         00048200
        DELTI := DTIME / 2.0 ;                                           00048300
        FOR  J := 2 STEP 1 UNTIL NYENDM  DO                              00048400
          BEGIN                                                          00048500
          SMDT[J] := RMOS[J] / DELTI ;                                   00048600
          EE[J] := SMDT[J] + RERX4 ;                                     00048700
          ES[J] := SMDT[J] + RERY4 ;                                     00048800
          RE[J] := SMDT[J] - RERY4 ;                                     00048900
          RS[J] := SMDT[J] - RERX4 ;                                     00049000
          END ;                                                          00049100
        SETPT1 ;                                                         00049200
        END ;                                                            00049300
%****************************************************************************00049400
%                                                                        00049500
    PROCEDURE THOMAS ( K1, E , TZETA ) ;                                 00049600
                        % THE THOMAS METHOD FOR INVERTING                00049700
                        %    TRIDINGONAL MATRICIES                       00049800
        DOUBLE ARRAY  E[*] , TZETA[*] ;                                  00049900
        INTEGER K1 ;                                                     00050000
        BEGIN                                                            00050100
        DOUBLE ARRAY WR[0:ZD ], VR[0:ZD ], RR[0:ZD ]  ;                  00050200
        INTEGER I, J, K ,L  , K2 ;                                       00050300
        WR[2] := D[2] / E[2] ;                                           00050400
        VR[2] := G[2] / E[2] ;                                           00050500
        FOR K := 3 STEP 1 UNTIL K1   DO                                  00050600
          BEGIN                                                          00050700
          RR[K] := E[K] - F[K] * WR[K-1]   ;                             00050800
          WR[K] := D[K] / RR[K] ;                                        00050900
          END ;                                                          00051000
        FOR K := 3 STEP 1 UNTIL K1  DO                                   00051100
          BEGIN                                                          00051200
          VR[K] := (G[K] - F[K]*VR[K-1] ) / RR[K] ;                      00051300
          END ;                                                          00051400
        L := K1 ;                                                        00051500
        TZETA[L] := VR[L] ;                                              00051600
        K2 := K1 -2 ;                                                    00051700
        FOR  K := 1 STEP 1 UNTIL K2  DO                                  00051800
          BEGIN                                                          00051900
          L := K1 - K ;                                                  00052000
          TZETA[L] := VR[L] - WR[L] * TZETA[L+1]  ;                      00052100
          END ;                                                          00052200
        END ;                                                            00052300
%****************************************************************************00052400
%                                                                        00052500
    PROCEDURE VORTE      ;                                               00052600
                        % FIRST HALF OF ADI                              00052700
        BEGIN                                                            00052800
        DOUBLE ARRAY TE[0:XD ] , TZETA[0:XD] ;                           00052900
        DOUBLE PFR1, PFR2, Q, S  ;                                       00053000
        INTEGER I, J, K ,L  ;                                            00053100
        FOR  I := 2 STEP 1 UNTIL NXENDM  DO                              00053200
          BEGIN                                                          00053300
```

```
          TZETA[I] := ZETA[I,1] ;                                          00053400
          END ;                                                            00053500
     FOR  J := 2 STEP 1 UNTIL NYENDM  DO                                   00053600
          BEGIN                                                            00053700
          FOR I := 2 STEP 1 UNTIL NXENDM  DO                               00053800
               BEGIN                                                       00053900
               PFR1 := -RDS4 * ( PSI[I+1,J] - PSI[I-1,J] ) ;               00054000
               PFR2 := -RDS4 * ( PSI[I,J+1] - PSI[I,J-1] ) ;               00054100
               F[I] := PFR2 - RERX2 ;                                      00054200
               TE[I] := EE[J] ;                                            00054300
               D[I]  := - PFR2 - RERX2 ;                                   00054400
               Q := RERY2 + PFR1 ;                                         00054500
               S := RERY2 - PFR1 ;                                         00054600
               G[I] := Q * ZETA[I,J-1] + RE[J] * ZETA[I,J]                 00054700
                    + S * ZETA[I,J+1] ;                                    00054800
               ZETA[I,J-1] := TZETA[I] ;                                   00054900
               END  ;                                                      00055000
          THOMAS ( NXENDM,TE , TZETA ) :                                   00055100
          END;                                                             00055200
     FOR  I:= 2 STEP 1 UNTIL NXENDM  DO                                    00055300
          BEGIN                                                            00055400
          ZETA[I,NYENDM] := TZETA[I] ;                                     00055500
          END ;                                                            00055600
     END;                                                                  00055700
%*****************************************************************************00055800
%                                                                           00055900
  PROCEDURE VORTS    :                                                    00056000
                     % SECOND HALF OF ADI                                  00056100
     BEGIN                                                                 00056200
     DOUBLE ARRAY TZETA[0:YD] ;                                            00056300
     DOUBLE PFR1, PFR2, Q, S  ;                                            00056400
     INTEGER I, J, K ,L  ;                                                 00056500
     FOR  J := 2 STEP 1 UNTIL NYENDM   DO                                  00056600
          BEGIN                                                            00056700
          TZETA[J] := ZETA[1,J] ;                                          00056800
          END;                                                             00056900
     FOR  I := 2 STEP 1 UNTIL NXENDM  DO                                   00057000
          BEGIN                                                            00057100
          FOR J := 2 STEP 1 UNTIL NYENDM  DO                               00057200
               BEGIN                                                       00057300
               PFR1 := -RDS4 * (PSI[I+1,J] - PSI[I-1,J] ) ;                00057400
               PFR2 := -RDS4 * (PSI[I,J+1] - PSI[I,J-1] ) ;                00057500
               F[J] := -PFR1 - RERY2 ;                                     00057600
               D[J] :=  PFR1 - RERY2 ;                                     00057700
               Q := RERX2 - PFR2 ;                                         00057800
               S := RERX2 + PFR2 ;                                         00057900
               G[J] := Q * ZETA[I-1,J] + RS[J] * ZETA[I,J]                 00058000
                    + S * ZETA[I+1,J] ;                                    00058100
               ZETA[I-1,J] := TZETA[J] ;                                   00058200
               END ;                                                       00058300
          G[2] := G[2] - F[2] * ZETA[I,1] ;                                00058400
          THOMAS  ( NYENDM, ES , TZETA ) :                                 00058500
          END ;                                                            00058600
     FOR  J := 2 STEP 1 UNTIL NYENDM  DO                                   00058700
          BEGIN                                                            00058800
          ZETA[NXENDM,J] := TZETA[J] ;                                     00058900
          END ;                                                            00059000
     END ;                                                                 00059100
%*****************************************************************************00059200
%                                                                           00059300
```

```
   PROCEDURE  TFOLD(IS,L,Z) ;                                                00059400
        INTEGER IS,L ;                                                       00059500
        DOUBLE ARRAY  Z[*]  ;                                                00059600
        BEGIN                                                                00059700
        INTEGER  I,I1,I2,IH2 ;                                               00059800
        DOUBLE A ;                                                           00059900
        IH2 := N2/2 - 1 ;                                                    00060000
        FOR I:=IS STEP 1 UNTIL IH2  DO                                       00060100
             BEGIN                                                           00060200
             I1 := I+L ;                                                     00060300
             I2 := N2-I+L ;                                                  00060400
             A := Z[I1] ;                                                    00060500
             Z[I1] := A - Z[I2] ;                                            00060600
             Z[I2] := A + Z[I2] ;                                            00060700
             END ;                                                           00060800
        END ;                                                                00060900
%************************************************************************   00061000
%                                                                            00061100
   PROCEDURE  ZERO(L) ;                                                      00061200
        INTEGER  L ;                                                         00061300
        BEGIN                                                                00061400
        INTEGER I ;                                                          00061500
        FOR  I := 1 STEP 1 UNTIL N2  DO                                      00061600
             BEGIN                                                           00061700
             Z[L+I-1] := 0.0 ;                                               00061800
             END ;                                                           00061900
        END ;                                                                00062000
%************************************************************************   00062100
%                                                                            00062200
   PROCEDURE KFOLD ;                                                         00062300
        BEGIN                                                                00062400
        INTEGER JS1,I,J5,IS1,IS0,IC1,IC0,J3,K1 ;                             00062500
        LABEL  L302,L301,L300,L402,L502,L404,L405,L304,L305,L505,L900 ;       00062600
        DOUBLE  SN,CS,ODD1,ODD2 ;                                            00062700
        JS1 := N2 ;                                                          00062800
        I := 1 ;                                                             00062900
        J5 := ISL + N2 ;                                                     00063000
        IS1 := ISL ;                                                         00063100
        IC1 := L1 ;                                                          00063200
        JS1 := JS1/2 ;                                                       00063300
        IF JS1 EQL 1 THEN  GO TO  L404 ;                                     00063400
        SN := SI[I] ;                                                        00063500
        IS1 := IS1+JS1 ;                                                     00063600
        IC1 := IC1+JS1 ;                                                     00063700
        J3 := IS1 + JS1 ;                                                    00063800
 L302:  IS0 := IS1-JS1 ;                                                     00063900
        IC0 := IC1-JS1 ;                                                     00064000
        ODD1 := SN*(Z[IC1]-Z[IS1]) ;                                         00064100
        ODD2 := SN*(Z[IC1]+Z[IS1]) ;                                         00064200
        Z[IC1] := Z[IC0]-ODD1 ;                                              00064300
        Z[IC0] := Z[IC0]+ODD1 ;                                              00064400
        Z[IS1] := -Z[IS0] + ODD2 ;                                           00064500
        Z[IS0] := Z[IS0] + ODD2 ;                                            00064600
        IS1 := IS1 + 1 ;                                                     00064700
        IC1 := IC1 + 1 ;                                                     00064800
        IF IS1 NEQ J3 THEN GO TO L302 ;                                      00064900
        I := I+1 ;                                                           00065000
 L301:  IS1 := ISL ;                                                         00065100
        IC1 := L1 ;                                                          00065200
        JS1 := JS1/2 ;                                                       00065300
```

```
%                                                                    00065400
        IF JS1 EQL 1 THEN GO TO L304 ;                               00065500
 L300:  SN := SI[I] ;                                                00065600
        I := I+1 ;                                                   00065700
        CS := SI[I] ;                                                00065800
        IS1 := IS1+JS1 ;                                             00065900
        IC1 := IC1 + JS1 ;                                           00066000
        J3 := IS1+JS1 ;                                              00066100
 L402:  IS0 := IS1-JS1 ;                                             00066200
        IC0 := IC1-JS1 ;                                             00066300
        ODD1 := CS*Z[IC1] - SN*Z[IS1] ;                              00066400
        ODD2 := SN*Z[IC1] + CS*Z[IS1] ;                              00066500
        Z[IC1] := Z[IC0] - ODD1 ;                                    00066600
        Z[IC0] := Z[IC0] + ODD1 ;                                    00066700
        Z[IS1] := -Z[IS0] + ODD2 ;                                   00066800
        Z[IS0] := Z[IS0] + ODD2 ;                                    00066900
        IS1 := IS1 + 1 ;                                             00067000
        IC1 := IC1 + 1 ;                                             00067100
        IF IS1 NEQ J3 THEN GO TO L402 ;                              00067200
        IS1 := IS1 +JS1 ;                                            00067300
        IC1 := IC1 +JS1 ;                                            00067400
        J3 := IS1 + JS1 ;                                            00067500
 L502:  IS0 := IS1 - JS1 ;                                           00067600
        IC0 := IC1 - JS1 ;                                           00067700
        ODD1 := SN*Z[IC1] - CS*Z[IS1] ;                              00067800
        ODD2 := CS*Z[IC1] + SN*Z[IS1] ;                              00067900
        Z[IC1] := Z[IC0] - ODD1 ;                                    00068000
        Z[IC0] := Z[IC0] + ODD1 ;                                    00068100
        Z[IS1] := -Z[IS0] + ODD2 ;                                   00068200
        Z[IS0] := Z[IS0] + ODD2 ;                                    00068300
        IS1 := IS1 + 1 ;                                             00068400
        IC1 := IC1 + 1 ;                                             00068500
        IF IS1 NEQ J3  THEN GO TO L502 ;                             00068600
        I := I+1 ;                                                   00068700
        IF  IS1 EQL J5  THEN GO TO  L301 ;                           00068800
        GO TO L300 ;                                                 00068900
%                                                                    00069000
 L404:  K1 := INDEX[I] ;                                             00069100
        SN := SI[I] ;                                                00069200
        IS0 := IS1 ;                                                 00069300
        IS1 := IS1 + JS1 ;                                           00069400
        IC0 := IC1 ;                                                 00069500
        IC1 := IC1 + JS1 ;                                           00069600
        ODD1 := SN*(Z[IC1] - Z[IS1]) ;                               00069700
        Y[K1+1] := Z[IC0] + ODD1 ;                                   00069800
        Y[N3-K1+1] := Z[IC0] - ODD1 ;                                00069900
 L405:  GO TO L900 ;                                                 00070000
%                                                                    00070100
 L304:  K1 := INDEX[I] ;                                             00070200
        SN := SI[I] ;                                                00070300
```

```
        I := I + 1 ;                                                    00070400
        CS := SI[I] ;                                                   00070500
        IS0 := IS1 ;                                                    00070600
        IS1 := IS1 + JS1 ;                                              00070700
        IC0 := IC1 ;                                                    00070800
        IC1 := IC1 + JS1 ;                                              00070900
        ODD1 := CS*Z[IC1] - SN*Z[IS1] ;                                 00071000
        Y[K1+1] := Z[IC0] + ODD1 ;                                      00071100
        Y[N3-K1+1] := Z[IC0] - ODD1 ;                                   00071200
%                                                                       00071300
 L305:  IS1 := IS1 +1  ;                                                00071400
        IC1 := IC1 +1  ;                                                00071500
        K1 := INDEX[I] ;                                                00071600
        IS0 := IS1 ;                                                    00071700
        IS1 := IS1 + JS1 ;                                              00071800
        IC0 := IC1 ;                                                    00071900
        IC1 := IC1 + JS1 ;                                              00072000
        ODD1 := SN*Z[IC1] - CS*Z[IS1] ;                                 00072100
        Y[K1+1] := Z[IC0] + ODD1 ;                                      00072200
        Y[N3-K1+1] := Z[IC0] - ODD1 ;                                   00072300
 L505:  IS1 := IS1 + 1 ;                                                00072400
        IC1 := IC1 + 1 ;                                                00072500
        I := I+1 ;                                                      00072600
        IF IS1 NEQ J5 THEN GO TO L304 ;                                 00072700
 L900:  END ;                                                           00072800
%***********************************************************************00072900
%                                                                       00073000
   PROCEDURE NEG(I1,I3,I2) ;                                            00073100
        INTEGER I1,I3,I2 ;                                              00073200
        BEGIN                                                           00073300
        INTEGER K ;                                                     00073400
        FOR K:=I1 STEP I2 UNTIL I3  DO                                  00073500
             BEGIN                                                      00073600
             Y[K] := -Y[K] ;                                            00073700
             END ;                                                      00073800
        END ;                                                           00073900
%***********************************************************************00074000
%                                                                       00074100
   PROCEDURE  FOUR67(IQ) ;                                              00074200
        INTEGER IQ ;                                                    00074300
        BEGIN                                                           00074400
        INTEGER N5,N11,N31,I,IP,JF,I1,I2,I3 ;                           00074500
        N4 := 2**IQ ;                                                   00074600
        N3 := N4 ;                                                      00074700
        N5 := N3/4 ;                                                    00074800
        N7 := N3/2 ;                                                    00074900
        N11 := 3*N7 ;                                                   00075000
        N31 := N3+1 ;                                                   00075100
        Z[N31] := 0.0 ;                                                 00075200
        Z[1] := 0.0 ;                                                   00075300
        N2 := N3 ;                                                      00075400
        FOR I:= 2 STEP 1 UNTIL IQ   DO                                  00075500
             BEGIN                                                      00075600
             TFOLD(1,1,Z) ;                                             00075700
             N2 := N2/2 ;                                               00075800
             END ;                                                      00075900
        Y[N7+1] := Z[2] ;                                               00076000
        JF := N5 ;                                                      00076100
        FOR IP := 2 STEP 1 UNTIL IQ  DO                                 00076200
             BEGIN                                                      00076300
```

```
            L1 := N2+1 ;                                                    00076400
            ZERO(1) ;                                                       00076500
            ISL := 1 ;                                                      00076600
            KFOLD ;                                                         00076700
            I1 := 3*JF+1 ;                                                  00076800
            I2 := 4*JF ;                                                    00076900
            I3 := I1+(N2/2 -1)*I2 ;                                         00077000
            NEG(I1,I3,I2) ;                                                 00077100
            N2 := N2+N2 ;                                                   00077200
            JF := JF/2 ;                                                    00077300
            END ;                                                           00077400
      END ;                                                                 00077500
%*********************************************************************     00077600
%                                                                         00077700
   PROCEDURE  RHSE ;                                                      00077800
      BEGIN                                                               00077900
      DOUBLE W,X,B,F1,F2 ;                                                00078000
      B := 1.0/HXY2 ;                                                     00078100
      F1 := B * B ;                                                       00078200
      F2 := 2.0 * B * (B+1) ;                                             00078300
      FOR J:= 3 STEP 2 UNTIL NYENDM2 DO                                   00078400
         BEGIN                                                            00078500
         X := 0.0 ;                                                       00078600
         X := -F1*(X+KORE[3,J])+F2*KORE[2,J]+B*(KORE[2,J+1]               00078700
                +KORE[2,J-1] )  ;                                         00078800
         FOR I:=3  STEP 1 UNTIL NXENDM2 DO                                00078900
            BEGIN                                                         00079000
            W := -F1*(KORE[I-1,J]+KORE[I+1,J])+F2*KORE[I,J]               00079100
                +B*(KORE[I,J+1]+KORE[I,J-1]) ;                            00079200
            KORE[I-1,J] := X ;                                            00079300
            X := W ;                                                      00079400
            END ;                                                         00079500
         W := 0.0 ;                                                       00079600
         W := -F1*(KORE[NXENDM2,J]+W)+F2*KORE[NXENDM,J]                   00079700
                +B*(KORE[NXENDM,J+1]+KORE[NXENDM,J-1]);                   00079800
         KORE[NXENDM2,J] := X ;                                           00079900
         KORE[NXENDM,J] := W ;                                            00080000
         END ;                                                            00080100
      END ;                                                               00080200
%*********************************************************************     00080300
%                                                                         00080400
   PROCEDURE  FETCHX(J) ;                                                 00080500
      INTEGER J ;                                                         00080600
      BEGIN                                                               00080700
      INTEGER I ;                                                         00080800
      FOR I:=  2   STEP 1 UNTIL NXENDM DO                                 00080900
         BEGIN                                                            00081000
         Z[I] := KORE[I,J] :                                              00081100
         END ;                                                            00081200
      END ;                                                               00081300
%*********************************************************************     00081400
%                                                                         00081500
   PROCEDURE  STOREX(J) ;                                                 00081600
      INTEGER J ;                                                         00081700
      BEGIN                                                               00081800
      INTEGER I ;                                                         00081900
      FOR I:= 2   STEP 1 UNTIL NXENDM DO                                  00082000
         BEGIN                                                            00082100
         KORE[I,J] := Y[I] ;                                              00082200
         END ;                                                            00082300
```

```
        END ;                                                                   00082400
%***************************************************************************   00082500
%                                                                               00082600
   PROCEDURE  CREDI(I,A,IP1) ;                                                  00082700
        INTEGER I,IP1 ;                                                         00082800
        DOUBLE A ;                                                              00082900
        BEGIN                                                                   00083000
        LABEL  L21,L2,L12,L25 ;                                                 00083100
        DOUBLE ARRAY BB[0:11] ;                                                 00083200
        INTEGER N1,NN,J,NV,NW,NU,IP ;                                           00083300
        DOUBLE B ;                                                              00083400
        IP := IP1 ;                                                             00083500
        BB[1] := A ;                                                            00083600
        B := A ;                                                                00083700
        N4 := 0 ;                                                               00083800
        IP := IP-1 ;                                                            00083900
        FOR N1 :=1 STEP 1 UNTIL IP  DO                                          00084000
             BEGIN                                                              00084100
             N4 := N4 + 1 ;                                                     00084200
             NW := 2**N1 ;                                                      00084300
             NV := 2*NW ;                                                       00084400
             IF (NV+1) GTR (NYEND-NV) THEN GO TO L21 ;                          00084500
             FOR J :=(NV+1) STEP NV UNTIL (NYEND-NV) DO                         00084600
                  BEGIN                                                         00084700
                  KORE[I,J] := B*KORE[I,J]+KORE[I,J-NW]+KORE[I,J+NW] ;          00084800
                  END ;                                                         00084900
 L21:        B := B*B - 2.0 ;                                                   00085000
             BB[N1+1] := B ;                                                    00085100
             IF B LSS 1.0@@14 THEN GO TO L2 ;                                   00085200
             IF (NV+1) GTR (NYEND-NV) THEN GO TO L12 ;                          00085300
             FOR J:=(NV+1) STEP NV UNTIL (NYEND-NV) DO                          00085400
                  BEGIN                                                         00085500
                  KORE[I,J] := -KORE[I,J]/B ;                                   00085600
                  END ;                                                         00085700
             GO TO L12 ;                                                        00085800
 L2:         END ;                                                              00085900
        J := (NYEND+1)/2 ;                                                      00086000
        KORE[I,J] := -KORE[I,J]/B ;                                             00086100
 L12:   FOR NN:=1 STEP 1 UNTIL N4  DO                                           00086200
             BEGIN                                                              00086300
             N1 := N4-NN ;                                                      00086400
             B := BB[N1+1] ;                                                    00086500
             NW := 2**N1 ;                                                      00086600
             NV := 2*NW ;                                                       00086700
             NU := 3*NV ;                                                       00086800
             J := NV+1 ;                                                        00086900
             KORE[I,J] := (KORE[I,J+NV] - KORE[I,J] )/B ;                       00087000
             J := NYEND-NV ;                                                    00087100
             KORE[I,J] := (KORE[I,J-NV] - KORE[I,J])/B ;                        00087200
             IF (NU+1) GTR (NYEND-NU) THEN GO TO L25 ;                          00087300
             FOR J:=(NU+1) STEP (2*NV) UNTIL (NYEND-NU) DO                      00087400
                  BEGIN                                                         00087500
                  KORE[I,J] := (KORE[I,J-NV]+KORE[I,J+NV]-KORE[I,J])/B:         00087600
                  END ;                                                         00087700
 L25:        END ;                                                              00087800
        END ;                                                                   00087900
%***************************************************************************   00088000
%                                                                               00088100
   PROCEDURE  CREDJ(J,A,IP1) ;                                                  00088200
        INTEGER J, IP1 ;                                                        00088300
```

```
        DOUBLE A ;                                                          00088400
        BEGIN                                                               00088500
        LABEL  L21,L2,L12,L25 ;                                             00088600
        DOUBLE ARRAY BB[0:11] ;                                             00088700
        INTEGER N1,NN,I,NV,NW,NU,IP ;                                       00088800
        DOUBLE B ;                                                          00088900
        IP := IP1 ;                                                         00089000
        BB[1] := A ;                                                        00089100
        B := A ;                                                            00089200
        N4 := 0 ;                                                           00089300
        IP := IP-1 ;                                                        00089400
        FOR N1 :=1 STEP 1 UNTIL IP  DO                                      00089500
             BEGIN                                                          00089600
             N4 := N4 + 1 ;                                                 00089700
             NW := 2**(N1-1) ;                                              00089800
             NV := 2*NW ;                                                   00089900
             IF (NV+1) GTR (NXEND-NV) THEN GO TO L21 ;                      00090000
             FOR I :=(NV+1) STEP NV UNTIL (NXEND-NV) DO                     00090100
                  BEGIN                                                     00090200
                  KORE[I,J] := B*KORE[I,J]+KORE[I-NW,J]+KORE[I+NW,J] ;      00090300
                  END ;                                                     00090400
L21:         B := B*B - 2.0 ;                                               00090500
             BB[N1+1] := B ;                                                00090600
             IF B LSS 1.0@@14 THEN GO TO L2 ;                               00090700
             IF (NV+1) GTR (NXEND-NV) THEN GO TO L12 ;                      00090800
             FOR I:=(NV+1) STEP NV UNTIL (NXEND-NV) DO                      00090900
                  BEGIN                                                     00091000
                  KORE[I,J] := -KORE[I,J]/B ;                               00091100
                  END ;                                                     00091200
             GO TO L12 ;                                                    00091300
L2:          END ;                                                          00091400
        I := (NXEND+1)/2 ;                                                  00091500
        KORE[I,J] := -KORE[I,J]/B ;                                         00091600
L12:    FOR NN:=1 STEP 1 UNTIL N4  DO                                       00091700
             BEGIN                                                          00091800
             N1 := N4-NN ;                                                  00091900
             B := BB[N1+1] ;                                                00092000
             NV := 2**N1 ;                                                  00092100
             NU := 3*NV ;                                                   00092200
             I := NV+1 ;                                                    00092300
             KORE[I,J] := (KORE[I+NV,J] - KORE[I,J] )/B ;                   00092400
             I := NXEND-NV ;                                                00092500
             KORE[I,J] := (KORE[I-NV,J] - KORE[I,J])/B ;                    00092600
             IF (NU+1) GTR (NXEND-NU) THEN GO TO L25 ;                      00092700
             FOR I:=(NU+1) STEP (2*NV) UNTIL (NXEND-NU) DO                  00092800
                  BEGIN                                                     00092900
                  KORE[I,J] := (KORE[I-NV,J]+KORE[I+NV,J]-KORE[I,J])/B;     00093000
                  END ;                                                     00093100
L25:         END ;                                                          00093200
        END ;                                                               00093300
%***************************************************************************00093400
%                                                                           00093500
   PROCEDURE  RHSO ;                                                        00093600
        BEGIN                                                               00093700
        INTEGER J,I  ;                                                      00093800
```

```
        DOUBLE F1,F2 ;                                                  00093900
        F1 := 1.0 / POTFAC ;                                            00094000
        F2 := HXY2 ;                                                    00094100
        FOR J:=2 STEP 2 UNTIL NYENDM DO                                 00094200
             BEGIN                                                      00094300
             FOR I:= 2   STEP 1 UNTIL NXENDM DO                         00094400
                  BEGIN                                                 00094500
                  KORE[I,J]:=F1*KORE[I,J]-F2*(KORE[I,J+1]+KORE[I,J-1]);00094600
                  END ;                                                 00094700
             END ;                                                      00094800
        END ;                                                           00094900
%***************************************************************      00095000
%                                                                       00095100
   PROCEDURE  POT1 ;                                                    00095200
        BEGIN                                                           00095300
        INTEGER  I,J ;                                                  00095400
        DOUBLE  A ;                                                     00095500
        FOR J:=  2   STEP 1 UNTIL NYENDM DO                             00095600
             BEGIN                                                      00095700
             KORE[2,J] := KORE[2,J]-POTFAC*KORE[1,J] ;                  00095800
             KORE[NXENDM,J] := KORE[NXENDM,J]-POTFAC*KORE[NXEND,J] ;    00095900
             END;                                                       00096000
%                                                                       00096100
        RHSE ;                                                          00096200
        FOR I:= 2   STEP 1 UNTIL NXENDM DO                              00096300
             BEGIN                                                      00096400
             KORE[I,3] := KORE[I,3]-POTFAC*KORE[I,1] ;                  00096500
             KORE[I,NYENDM2] := KORE[I,NYENDM2]-POTFAC*KORE[I,NYEND] ;  00096600
             END ;                                                      00096700
        FOR J:= 3 STEP 2 UNTIL NYENDM2 DO                               00096800
             BEGIN                                                      00096900
             FETCHX(J) ;                                                00097000
             FOUR67(IQX) ;                                              00097100
             STOREX(J) ;                                                00097200
             END ;                                                      00097300
        FOR I:= 2   STEP 1 UNTIL NXENDM DO                              00097400
             BEGIN                                                      00097500
             A := AKX[I] ;                                              00097600
             CREDI(I,A,IQY-1) ;                                         00097700
             END ;                                                      00097800
        FOR J:= 3 STEP 2 UNTIL NYENDM2 DO                               00097900
             BEGIN                                                      00098000
             FETCHX(J) ;                                                00098100
             FOUR67(IQX) ;                                              00098200
             STOREX(J) ;                                                00098300
             END ;                                                      00098400
        RHSO ;                                                          00098500
        FOR J:=2 STEP 2 UNTIL NYENDM DO                                 00098600
             BEGIN                                                      00098700
             A := 2.0 *(1 + HXY2) ;                                     00098800
             CREDJ(J,A,IQX) ;                                           00098900
             END ;                                                      00099000
        END ;                                                           00099100
%***************************************************************      00099200
%                                                                       00099300
   PROCEDURE SETUP ;                                                    00099400
        BEGIN                                                           00099500
        INTEGER L,I,J ;                                                 00099600
        FOR J := 2 STEP 1 UNTIL NYENDM DO                               00099700
             BEGIN                                                      00099800
```

```
            FOR I := 2 STEP 1 UNTIL NXENDM DO                                   00099900
                 BEGIN                                                          00100000
                 KORE[I,J] := ZETA[I,J] * RMOS[J] * (DELX**2) * POTFAC00100100
                 END ;                                                          00100200
            END ;                                                               00100300
       END ;                                                                    00100400
%***********************************************************************      00100500
%                                                                               00100600
   PROCEDURE SETDOWN ;                                                          00100700
       BEGIN                                                                    00100800
       INTEGER L,I,J ;                                                          00100900
       FOR J := 2 STEP 1 UNTIL NYENDM DO                                        00101000
            BEGIN                                                               00101100
            FOR I := 2 STEP 1 UNTIL NXENDM DO                                   00101200
                 BEGIN                                                          00101300
                 PSI[I,J] := KORE[I,J];                                         00101400
                 END ;                                                          00101500
            END ;                                                               00101600
       END ;                                                                    00101700
%***********************************************************************      00101800
%                                                                               00101900
   PROCEDURE STREAM ;                                                           00102000
       BEGIN                                                                    00102100
       SETUP ;                                                                  00102200
       POT1 ;                                                                   00102300
       SETDOWN ;                                                                00102400
       END;                                                                     00102500
%***********************************************************************      00102600
%                                                                               00102700
   PROCEDURE  SURFACEVORTICITY ;                                                00102800
                    % COMPUTE THE SURFACE VORTICITY                             00102900
       BEGIN                                                                    00103000
       INTEGER I, J, K ,L  ;                                                    00103100
       FOR  I := 2 STEP 1 UNTIL NXENDM  DO                                      00103200
            BEGIN                                                               00103300
            ZETA[I,1]:=(8.0 * PSI[I,2] - PSI[I,3] ) * WFBZ2 ;                   00103400
            END ;                                                               00103500
       END ;                                                                    00103600
%***********************************************************************      00103700
%                                                                               00103800
   PROCEDURE SHORTTIMESOLUTION ( T ) ;                                          00103900
       DOUBLE T ;                                                               00104000
       BEGIN                                                                    00104100
       DOUBLE ARRAY S[0:XD] , C[0:XD], FDZT[0:XD] ;                             00104200
       DOUBLE  VAR1,VAR2,VAR3,VAR4,VAR5,VAR6,ALF,CST1,CST2,CST3,CST4,           00104300
                    CST5,CST6,CST7,CST8,CST9,CST10,CST11,CST12,                 00104400
                    CST13,CST14,CST15,CST16,CST17,CST18,CST19,                  00104500
                    CST20,CST21,CST22,CST23,CST24,VAR7,CST25,CST26,             00104600
                    INT , TSI, ETA , DTSI, LEVEL,                               00104700
                    VAR8,VAR9,VAR12,VAR13,                                      00104800
                    VAR10, CST27, CST28, CST29, CST30, VAR11,                   00104900
                    VAR14,PITSI,VAR15,VAR16,VAR17,VAR18,                        00105000
                    VAR19,VAR20,VAR21,VAR22,VAR23,CST31,VAR24 ;                 00105100
       CST1 := DSQRT(PI) ;                                                      00105200
       CST2 := 1.0/CST1 ;                                                       00105300
       CST3 := 1.0 / PI ;                                                       00105400
       CST4 := PI * PI ;                                                        00105500
       CST5 := CST3 * CST3 ;                                                    00105600
       CST6 := DSQRT(2.0);                                                      00105700
       CST7 := 1.0 / CST6 ;                                                     00105800
```

```
CST8 := 1.0 / 3.0 ;                                                    00105900
CST9 := 4.0 * CST2 ;                                                   00106000
CST10 := 0.25 * CST1 ;                                                 00106100
CST11 := 1.5 * CST1 ;                                                  00106200
CST12 := CST8 * CST3 ;                                                 00106300
CST13 := 0.5 - 2.0 * CST12 ;                                           00106400
CST14 := 2.0 * CST8 * CST2 ;                                           00106500
CST15 := CST14 * CST14 ;                                               00106600
CST16 := 1.0 + CST15 ;                                                 00106700
CST17 := CST2 * CST16 ;                                                00106800
CST18 := 11.0 * CST2 / 6.0 ;                                           00106900
CST19 := CST2 * ( CST15 - 1.5 ) ;                                      00107000
CST20 := 8.0 * CST8 * CST7 * CST2 ;                                    00107100
CST21 := CST2 * ( 8.0 * CST8 * CST7 - CST15 - 1.0 ) ;                  00107200
CST22 := 2.0 * T / REYN ;                                              00107300
CST23 := DSQRT( CST22 ) ;                                              00107400
CST24 := 1.0 / CST23 ;                                                 00107500
CST25 := 0.5 * CST24 ;                                                 00107600
CST26 := 8.0 * CST8 * CST7 ;                                           00107700
CST27 := REYN /( 8.0 * T ) ;                                           00107800
CST28 := 4.0 * CST22 ;                                                 00107900
CST29 := DSQRT(CST28) ;                                                00108000
CST30 := 4.0 * CST12 ;                                                 00108100
CST31 := 2.0 * CST12 ;                                                 00108200
ETA := 0.0 ;                                                           00108300
FOR I := 1 STEP 1 UNTIL NXENDM/2 DO                                    00108400
     BEGIN                                                             00108500
     S[I] := DSIN(PI*ETA) ;                                            00108600
     C[I] := DCOS(PI*ETA) ;                                            00108700
     S[NXENDP-I] := S[I] ;                                             00108800
     C[NXENDP-I] := -C[I] ;                                            00108900
     ETA := ETA + DELX ;                                               00109000
     END ;                                                             00109100
S[NXENDP/2] := 1.0 ;                                                   00109200
C[NXENDP/2] := 0.0 ;                                                   00109300
TSI := 0.0 ;                                                           00109400
DTSI := DELY ;                                                         00109500
FOR J := 1 STEP  1 UNTIL NYENDM DO                                     00109600
     BEGIN                                                             00109700
     PITSI := PI * TSI ;                                               00109800
     VAR1 := DEXP ( PITSI ) ;                                          00109900
     VAR2 := 1.0 / VAR1 ;                                              00110000
     ALF := CST25 * (VAR1 - 1.0) ;                                     00110100
     IF TSI LSS 1.0@@-15 THEN                                          00110200
          BEGIN                                                        00110300
          ALF := CST25 * ( PITSI *                                     00110400
               ( 1.0 + 0.5 * PITSI * ( 1.0                             00110500
               + CST8 * PITSI ))) ;                                    00110600
          END ;                                                        00110700
     VAR3 := DEXP(-ALF * ALF ) ;                                       00110800
     VAR4 := ERF( ALF ) ;                                              00110900
     VAR5 := 1.0 - VAR4 ;                                              00111000
     VAR6 := ERF( CST6 * ALF ) ;                                       00111100
     VAR7 := 1.0 - VAR6 ;                                              00111200
     VAR11 := VAR2 * VAR2 ;                                            00111300
     VAR12 := CST29 * VAR2 ;                                           00111400
     VAR13 := 4.0 * CST28 * VAR11 ;                                    00111500
     VAR14 := VAR1 - VAR2 ;                                            00111600
     VAR17 := -VAR2 + VAR3 ;                                           00111700
     VAR18 := VAR3 * ( 11.0 * VAR5 - 9.0 ) + 4.0 * VAR5                00111800
```

```
          - 6.0 ;                                                    00111900
VAR19 := VAR3 - 1.0 ;                                                00112000
VAR21 := 1.0 + VAR11 - 2.0 * VAR5 ;                                  00112100
VAR22 := VAR11 - VAR3 ;                                              00112200
VAR23 := VAR5 - VAR3 ;                                               00112300
IF TSI LSS 1.0@@-15 THEN                                             00112400
     BEGIN                                                           00112500
     VAR14 := PITSI * ( 2.0 + CST8 * PITSI * PITSI ) ;               00112600
     VAR15 := ALF * ALF ;                                            00112700
     VAR16 := VAR15 * ( - 1.0 + 0.5 * VAR15 *                        00112800
          (1.0 - CST8 * VAR15 )) ;                                   00112900
     VAR20 := 2.0 * PITSI ;                                          00113000
     VAR24 := VAR20 * ( -1.0 + 0.5 * VAR20 *                         00113100
          ( 1.0 - CST8 * CST20 )) ;                                  00113200
     VAR21 := VAR24 + 2.0 * VAR4 ;                                   00113300
     IF ALF LSS 1.0@@-7 THEN                                         00113400
          BEGIN                                                      00113500
          VAR17 := PITSI * ( 1.0 - 0.5 * PITSI *                     00113600
               (1.0 + CST8 * PITSI )) + VAR16 ;                      00113700
          VAR22 := VAR21 - VAR16 ;                                   00113800
          END ;                                                      00113900
     END ;                                                           00114000
IF ALF LSS 1.0@@-7 THEN                                              00114100
     BEGIN                                                           00114200
     VAR15 := ALF * ALF ;                                            00114300
     VAR16 := VAR15 * ( - 1.0 + 0.5 * VAR15 *                        00114400
          (1.0 - CST8 * VAR15 )) ;                                   00114500
     VAR18 := 2.0 * VAR16 - VAR4 * ( 15.0 - 11.0 *                   00114600
          VAR16 ) ;                                                  00114700
     VAR19 := VAR16 ;                                                00114800
     VAR23 := - VAR4 - VAR16 ;                                       00114900
     END ;                                                           00115000
LEVEL :=   VAR14 + CST23 *  CST9 *                                   00115100
          ( VAR17 - CST1 * ALF * VAR5                                00115200
          + CST23 * ( -CST10 * VAR4 +                                00115300
          CST11 * ALF * ALF * VAR5 - 1.5 * ALF * VAR3 )) ;           00115400
INT := ALF * ( CST12 * VAR3 * VAR3 + VAR5 * ( 0.5 *                  00115500
          VAR4 - CST31) + ALF * ( VAR3 * (-CST14 * VAR5 +            00115600
          CST17) + ALF * VAR5 * (CST8 * VAR5 - CST16) ) )            00115700
          +CST2 * ( CST26 * VAR6 + CST15 *                           00115800
          VAR19 + 0.5 * CST8 * VAR18 ) ;                             00115900
VAR8 := 8.0 * CST23 * T * INT ;                                      00116000
VAR9 := VAR5 * ( 3.0 + VAR2 * ( 6.0 * CST23 * ALF                    00116100
          -2.0 + VAR2 * CST23 * ( ALF * ( 4.0 -                      00116200
          6.0 * CST23 * ALF ) - CST23 ))) + VAR3 *                   00116300
          2.0 * CST2 * ( -ALF + CST24 - 2.0 *                        00116400
          CST23 * VAR2 * ( 1.0 + VAR2 * ( 1.0 -                      00116500
          1.5 * CST23 * ALF ))) + CST22 * VAR2 *                     00116600
          VAR2 ;                                                     00116700
VAR10 := 4.0 * T * CST25 * ( VAR5 * ( VAR12 *                        00116800
          CST13 - VAR13 * CST14 + ALF * ( -6.0 *                     00116900
          CST16 - VAR13 * CST13 - VAR3 * VAR12 *                     00117000
          3.0 * CST2 + ALF * ( -3.0 * VAR12 *                        00117100
          CST16 + VAR3 * ( 2.0 * CST2 + VAR13 *                      00117200
          CST14 ) + ALF * ( VAR13 * CST16 -                          00117300
          VAR5 * VAR13 * CST8 ))) + VAR5 *                           00117400
          ( -0.5 * VAR12 +ALF * ( 2.0 + 0.5 *                        00117500
          VAR13 + ALF * VAR12 )) - VAR3 * ( CST2 +                   00117600
          CST18 * VAR13 )) + VAR3 * ( 2.0 * CST2 *                   00117700
```

```
                    ( 1.5 + CST30 ) - CST30 * VAR12 -                    00117800
                    VAR13 * CST19 + VAR3 * VAR12 * 2.0 *                 00117900
                    CST3 + ALF * ( 8.0 * CST12 + CST9 *                  00118000
                    VAR12 * ( 1.0 + CST12 ) - VAR3 * ( 2.0 *             00118100
                    CST3 + CST12 * VAR13 ) - ALF * CST2 *                00118200
                    ( 2.0 + VAR13 * CST16 ))) +                          00118300
                    VAR13 * CST20 * VAR7 - VAR13 * CST21 ) ;             00118400
          FOR I := 2 STEP 1 UNTIL NXENDM DO                              00118500
               BEGIN                                                     00118600
               PSI[I,J] := ( LEVEL + VAR8 * C[I] ) * S[I] ;              00118700
               ZETA[I,J] := ( VAR9 + VAR10 * C[I] ) * S[I];              00118800
               FDZT[I] := PI * ( 8.0 * CST2 * CST29 +  7.0 -             00118900
                    CST9 * CST25 + REYN * ( CST2 * CST29 *               00119000
                    ( 1.0 + CST30 ) - 2.0 ) * C[I] ) *                   00119100
                    S[I] ;                                               00119200
               SLOPE[I] := -FDZT[I] * CST3 ;                             00119300
               END ;                                                     00119400
          TSI := TSI + DTSI ;                                            00119500
          END ;                                                          00119600
     TIME1 := T ;                                                        00119700
     END ;                                                               00119800
%********************************************************************** 00119900
%                                                                       00120000
  PROCEDURE BC ;                                                        00120100
     BEGIN                                                              00120200
     INTEGER L,I,J;                                                     00120300
     FOR J := 1 STEP 1 UNTIL NYEND DO                                   00120400
          BEGIN                                                         00120500
          FOR I := 1 STEP 1 UNTIL NXEND DO                              00120600
               BEGIN                                                    00120700
               KORE[I,J] := PSI[I,J] ;                                  00120800
               END ;                                                    00120900
          END ;                                                         00121000
     END ;                                                              00121100
%*********************************************************************** 00121200
%                                                                       00121300
  PROCEDURE INITIALIZENET ;                                             00121400
                    % INITIALIZE THE PSI NET WITH THE POTENTIAL FLOW     00121500
                    %    SOLUTION, RELAX IT, COMPUTE THE SURFACE        00121600
                    %    VORTICITY,AND SET THE REST OF THE VORTICITY    00121700
                    %    NET EQUAL TO ZERO                              00121800
     BEGIN                                                              00121900
     INTEGER NXP, NXM, II , I,J,K,L ;                                   00122000
     REAL   TCG, ETCG, RETCGS, FTG, DIE, SI1, PIYIN, CSIGM,             00122100
                    PIDEL,  PIE ;                                       00122200
     LABEL L1 ;                                                         00122300
     IF SECTION NEQ 1 THEN BEGIN FETCHPSIZETA;BC;GO TO L1;END;          00122400
     NXM := NXEND / 2 ;                                                 00122500
     NXP := NXM + 1 ;                                                   00122600
     SIGMA := 0.0 ;                                                     00122700
     FOR  J := 1 STEP 1 UNTIL NYENDM  DO                                00122800
          BEGIN                                                         00122900
          TCG := PI * SIGMA ;                                           00123000
          ETCG := EXP(TCG) ;                                            00123100
          RETCGS := 1.0 / (ETCG * ETCG) ;                               00123200
          FTG := ETCG * (1.0 - RETCGS) ;                                00123300
          ETA := 0.0 ;                                                  00123400
          FOR I := 1 STEP 1 UNTIL NXP  DO                               00123500
               BEGIN                                                    00123600
               PIE := PI * ETA ;                                        00123700
```

```
                SI1 := SIN(PIE) ;                                          00123800
                PSI[I,J] := FTG * SI1 ;                                    00123900
                ETA := ETA + DELX ;                                        00124000
                END ;                                                      00124100
           SIGMA := SIGMA + DELY ;                                         00124200
           END;                                                            00124300
      PIYIN := PI * YINF ;                                                 00124400
      CSIGM := EXP(PIYIN) ;                                                00124500
      FTG := CSIGM ;                                                       00124600
      IF POTENTIAL=1 THEN                                                  00124700
           BEGIN                                                           00124800
           RETCGS := 1.0/(CSIGM*CSIGM) ;                                   00124900
           FTG := CSIGM * (1.0 - RETCGS) ;                                 00125000
           END ;                                                           00125100
      PIDEL := PI * DELX ;                                                 00125200
      FOR  I := 1 STEP 1 UNTIL NXP  DO                                     00125300
           BEGIN                                                           00125400
           FI := I-1 ;                                                     00125500
           PSI[I, NYEND] :=   FTG * SIN(PIDEL * FI) ;                      00125600
           END;                                                            00125700
      FOR J := 1 STEP 1 UNTIL NYEND   DO                                   00125800
           BEGIN                                                           00125900
           FOR I :=  1 STEP 1 UNTIL  NXM   DO                              00126000
                BEGIN                                                      00126100
                II := NXENDP - I ;                                         00126200
                PSI[II, J] := PSI[I,J] ;                                   00126300
                END ;                                                      00126400
           END ;                                                           00126500
      FOR J := 1 STEP 1 UNTIL NYEND   DO                                   00126600
           BEGIN                                                           00126700
           FOR I := 1 STEP 1 UNTIL NXEND  DO                               00126800
                BEGIN                                                      00126900
                ZETA[I,J] := 0.0 ;                                         00127000
                END ;                                                      00127100
           END ;                                                           00127200
      IF PP=1 THEN PRINT1 ( PSI , IPRNT ) ;                                00127300
      IF STARTTIME EQL 0.0 THEN BEGIN BC;STREAM;SURFACEVORTICITY;END;      00127400
L1:   IF STARTTIME GTR 1.0@@-30 AND                                        00127500
           SECTION EQL 1 THEN SHORTTIMESOLUTION(STARTTIME);                00127600
      END ;                                                                00127700
%**************************************************************************00127800
%                                                                          00127900
  PROCEDURE MATINV (A,N,B,M,DETERM ) ;                                     00128000
      DOUBLE ARRAY A[*,*] , B[*,*]  ;                                      00128100
      INTEGER N,M ;                                                        00128200
      DOUBLE DETERM  ;                                                     00128300
      BEGIN                                                                00128400
      DEFINE   IROW  = JROW #, ICOLUM  = JCOLUM  #, AMAX = T #,            00128500
                     SWAP = T # ;                                          00128600
      DOUBLE ARRAY  PIVOT[0:5] ;                                           00128700
      DOUBLE  T  ;                                                         00128800
      INTEGER  ARRAY IPIVOT[0:5],INDEX[0:5,0:2]  ;                         00128900
      INTEGER JROW, JCOLUM, L1 ,I,J,K,L ;                                  00129000
      LABEL HOP720 , HOP777 ;                                              00129100
      DETERM := 1.0 ;                                                      00129200
      FOR J := 1 STEP 1 UNTIL N  DO                                        00129300
           BEGIN                                                           00129400
           IPIVOT[J] := 0 ;                                                00129500
           END ;                                                           00129600
      FOR I := 1 STEP 1 UNTIL N  DO                                        00129700
```

```
          BEGIN                                                          00129800
          LABEL HOP260, HOP380, HOP550 ;                                 00129900
          AMAX := 0.0 ;                                                  00130000
          FOR J := 1 STEP 1 UNTIL N  DO                                  00130100
               BEGIN                                                     00130200
               LABEL HOP105  ;                                           00130300
               IF (IPIVOT[J]-1) EQL 0 THEN GO TO HOP105 ;                00130400
               FOR  K := 1 STEP 1 UNTIL N  DO                            00130500
                    BEGIN                                                00130600
                    LABEL HOP100 ;                                       00130700
                    IF (IPIVOT[K] -1) GTR 0 THEN GO TO HOP777 ;          00130800
                    IF (IPIVOT[K] -1) EQL 0 THEN GO TO HOP100 ;          00130900
                    IF (DABS(AMAX) - DABS(A[J,K] )) GTR 0.0 THEN         00131000
                                   GO TO HOP100 ;                        00131100
                    IROW := J ;                                          00131200
                    ICOLUM := K ;                                        00131300
                    AMAX := A[J,K] ;                                     00131400
HOP100:             END ;                                                00131500
HOP105:        END ;                                                     00131600
          IPIVOT[ICOLUM] := IPIVOT[ICOLUM] + 1 ;                         00131700
          IF (IROW - ICOLUM) EQL  0  THEN GO TO  HOP260 ;                00131800
          DETERM := - DETERM ;                                           00131900
          FOR  L := 1 STEP 1 UNTIL N  DO                                 00132000
               BEGIN                                                     00132100
               SWAP := A[IROW,L] ;                                       00132200
               A[IROW,L] := A[ICOLUM,L] ;                                00132300
               A[ICOLUM,L] := SWAP ;                                     00132400
               END ;                                                     00132500
          IF  M LEQ 0  THEN  GO TO  HOP260 ;                             00132600
          FOR L := 1 STEP 1 UNTIL M  DO                                  00132700
               BEGIN                                                     00132800
               SWAP := B[IROW,L ]  ;                                     00132900
               B[IROW,L] := B[ICOLUM,L] ;                                00133000
               B[ICOLUM,L] := SWAP ;                                     00133100
               END;                                                      00133200
HOP260:   INDEX[I,1] := IROW ;                                           00133300
          INDEX[I,2] := ICOLUM ;                                         00133400
          PIVOT[I]  := A[ICOLUM,ICOLUM] ;                                00133500
          DETERM := DETERM * PIVOT[I] ;                                  00133600
          IF PIVOT[I]  EQL 0  THEN GO TO  HOP720 ;                       00133700
          A[ICOLUM,ICOLUM] := 1.0 ;                                      00133800
          FOR  L := 1 STEP 1 UNTIL N  DO                                 00133900
               BEGIN                                                     00134000
               A[ICOLUM,L] := A[ICOLUM,L] / PIVOT[I]  ;                  00134100
               END ;                                                     00134200
          IF  M LEQ 0  THEN  GO TO HOP380  ;                             00134300
          FOR  L := 1 STEP 1 UNTIL M  DO                                 00134400
               BEGIN                                                     00134500
               B[ICOLUM,L] := B[ICOLUM,L] / PIVOT[I] ;                   00134600
               END ;                                                     00134700
HOP380:   FOR  L1 := 1 STEP 1 UNTIL  N   DO                              00134800
               BEGIN                                                     00134900
               IF (L1-ICOLUM)  EQL 0  THEN  GO TO HOP550 ;               00135000
               T := A[L1,ICOLUM]  ;                                      00135100
               A[L1,ICOLUM] := 0.0 ;                                     00135200
               FOR  L := 1 STEP 1 UNTIL N  DO                            00135300
                    BEGIN                                                00135400
                    A[L1,L] := A[L1,L] - A[ICOLUM,L] * T ;               00135500
                    END ;                                                00135600
               IF  M LEQ 0  THEN  GO TO HOP550 ;                         00135700
```

```
                    FOR  L :=  1 STEP 1 UNTIL  M  DO                          00135800
                         BEGIN                                                00135900
                         B[L1,L] := B[L1,L] - B[ICOLUM,L] *T ;                00136000
                         END ;                                                00136100
                    END;                                                      00136200
HOP550:        END  ;                                                         00136300
          FOR  I := 1 STEP 1 UNTIL N  DO                                      00136400
               BEGIN                                                          00136500
               LABEL HOP710 ;                                                 00136600
               L := N+ 1 - I ;                                                00136700
               IF (INDEX[L,1] - INDEX[L,2]) EQL 0 THEN GO TO HOP710;          00136800
               JROW := INDEX[L,1] ;                                           00136900
               JCOLUM := INDEX[L,2] ;                                         00137000
               FOR K := 1 STEP 1 UNTIL  N  DO                                 00137100
                    BEGIN                                                     00137200
                    SWAP := A[K,JROW] ;                                       00137300
                    A[K,JROW] := A[K,JCOLUM] ;                                00137400
                    A[K,JCOLUM] := SWAP ;                                     00137500
                    END ;                                                     00137600
HOP710:        GO TO HOP777;                                                  00137700
               END;                                                           00137800
HOP720:        BEGIN                                                          00137900
               FORMAT  FMT15 ("MATRIX IS SINGULAR" )   ;                      00138000
               WRITE (LINE, FMT15 )  ;                                        00138100
               GO TO HOPEND ;                                                 00138200
               END ;                                                          00138300
HOP777:        BEGIN                                                          00138400
               END ;                                                          00138500
          END ;                                                               00138600
%*************************************************************************** 00138700
%                                                                             00138800
   PROCEDURE  LSQ1 ( X,Y,W,N,L,A,B,M)  ;                                      00138900
          DOUBLE ARRAY  A[*,*] , B[*,*] ;                                     00139000
          INTEGER N,M,L ;                                                     00139100
          ARRAY  X[*], Y[*,*], W[*]   ;                                       00139200
          BEGIN                                                               00139300
          DOUBLE ARRAY C[0:5, 0:5]  ;                                         00139400
          DOUBLE DETERM ;                                                     00139500
          INTEGER I, J, K     ;                                               00139600
          FOR I := 1 STEP 1 UNTIL N   DO                                      00139700
               BEGIN                                                          00139800
               C[I,1] := 1.0 ;                                                00139900
               END ;                                                          00140000
          FOR J :=  2 STEP 1 UNTIL M  DO                                      00140100
               BEGIN                                                          00140200
               FOR I := 1 STEP 1 UNTIL N  DO                                  00140300
                    BEGIN                                                     00140400
                    C[I,J] := C[I,J-1] * X[I] ;                               00140500
                    END ;                                                     00140600
```

```
          END;                                                          00140700
       FOR I := 1 STEP 1 UNTIL M  DO                                    00140800
          BEGIN                                                         00140900
          FOR  J := 1 STEP 1 UNTIL M DO                                 00141000
               BEGIN                                                    00141100
               A[I,J] :=  0.0 ;                                         00141200
               FOR K := 1 STEP 1 UNTIL  N  DO                           00141300
                    BEGIN                                               00141400
                    A[I,J] := A[I,J] + C[K,I]*C[K,J] * W[K] ;           00141500
                    END ;                                               00141600
               END ;                                                    00141700
          END ;                                                         00141800
       FOR  I := 1 STEP 1 UNTIL M  DO                                   00141900
          BEGIN                                                         00142000
          B[I,1] := 0.0 ;                                               00142100
          FOR K := 1 STEP 1 UNTIL  N  DO                                00142200
               BEGIN                                                    00142300
               B[I,1] := B[I,1] + C[K,I] * Y[K,1]*W[K] ;                00142400
               END ;                                                    00142500
          END ;                                                         00142600
       MATINV ( A,M,B,L, DETERM ) ;                                     00142700
       END ;                                                            00142800
%***************************************************************************00142900
%                                                                       00143000
   PROCEDURE PRESS          ;                                           00143100
       BEGIN                                                            00143200
       REAL ARRAY RAD[0:5], W[0:5], AZETA[0:5,0:1], ZETAS[0:XD],        00143300
                  PRES[0:XD]  ;                                         00143400
       REAL PISIG, REDEL, THETA, PAREA, CD  , SAREA ;                   00143500
       INTEGER I, J, K ,L  ;                                            00143600
       LABEL L1 ;                                                       00143700
       FORMAT FMT12 ( X50,"PRESSURE DISTRIBUTION"// X45,"TIME =",       00143800
                  F7.3, X6, "NITER =", I5 // X50, "ANGLE", X9,          00143900
                  "PRESSURE"/ X50, "******", X9, "*********" //         00144000
                  ( X50, F5.1, X10, F7.3) ) ; % 104                     00144100
       FORMAT FMT13 ( //// X56,"CDP =", F9.3// X56,"CDF =",             00144200
                  F9.3 // X56,"CD  =",F9.3) ; % 109                     00144300
       FORMAT FMT72 ( X50, F5.1, X10, F7.3 ) ;                          00144400
       IF STARTTIME GTR 0.0 AND STARTTIME EQL TIME1 THEN GO TO L1;      00144500
       SIGMA := 0.0 ;                                                   00144600
       FOR  J := 1 STEP 1 UNTIL 5  DO                                   00144700
          BEGIN                                                         00144800
          PISIG := PI * SIGMA ;                                         00144900
          W[J] := 1.0 ;                                                 00145000
          RAD[J] := EXP(PISIG) ;                                        00145100
          SIGMA := SIGMA + DELY ;                                       00145200
          END ;                                                         00145300
       FOR I := 1 STEP 1 UNTIL NXEND  DO                                00145400
          BEGIN                                                         00145500
          FOR  J := 1 STEP 1 UNTIL  5  DO                               00145600
               BEGIN                                                    00145700
               AZETA[J,1]:= ZETA[I,J] ;                                 00145800
               END ;                                                    00145900
          LSQ1 ( RAD,AZETA,W,4,1,A,B,3 ) ;                              00146000
          SLOPE[I] := B[2,1] + 2.0 * B[3,1] ;                           00146100
          END;                                                          00146200
L1:    PRES[1] := 0.0 ;                                                 00146300
       REDEL := (4.0/REYN) * (PI* DELX/3.0) ;                           00146400
       FOR  I := 3 STEP 2 UNTIL NXEND DO                                00146500
          BEGIN                                                         00146600
```

```
          PRES[I] := PRES[I-2] - REDEL * (SLOPE[I-2] + 4.0 *          00146700
                     SLOPE[I-1] + SLOPE[I] ) ;                        00146800
          END ;                                                       00146900
        FOR  I := 1 STEP 2 UNTIL NXEND DO                             00147000
          BEGIN                                                       00147100
          FI := I-1 ;                                                 00147200
          THETA := FI * PI * DELX ;                                   00147300
          ZETAS[I] := -PRES[I] * COS(THETA) ;                         00147400
          END ;                                                       00147500
        PAREA := 0.0 ;                                                00147600
        L := 5 ;                                                      00147700
        IF NYENDM MOD 4 NEQ 0 THEN                                    00147800
          BEGIN                                                       00147900
          PAREA := 1.50 * (ZETAS[1] + ZETAS[3]) ;                     00148000
          L := 7 ;                                                    00148100
          END ;                                                       00148200
        FOR  I := L STEP 4 UNTIL NXEND DO                             00148300
          BEGIN                                                       00148400
          PAREA := PAREA + (ZETAS[I-4] + 4.0 * ZETAS[I-2]             00148500
                     +ZETAS[I] ) ;                                    00148600
          END ;                                                       00148700
        PAREA := (PI * 2.0 * DELX/3.0 ) * PAREA ;                     00148800
        ZETAS[1] := 0.0 ;                                             00148900
        FOR  I := 2 STEP 1 UNTIL NXEND  DO                            00149000
          BEGIN                                                       00149100
          FI := I-1 ;                                                 00149200
          THETA:= FI * PI * DELX ;                                    00149300
          ZETAS[I] := ZETA[I,1] * SIN(THETA) ;                        00149400
          END ;                                                       00149500
        SAREA := 0.0 ;                                                00149600
        FOR  I := 3 STEP 2 UNTIL NXEND  DO                            00149700
          BEGIN                                                       00149800
          SAREA := SAREA + ( ZETAS[I-2] + 4.0 * ZETAS[I-1]            00149900
                     + ZETAS[I] ) ;                                   00150000
          END ;                                                       00150100
        SAREA := -REDEL * SAREA ;                                     00150200
        CD := PAREA + SAREA ;                                         00150300
        WRITE ( LINE [SKIP 1 ] ) ;                                    00150400
        PGNUM := PGNUM + 1 ;                                          00150500
        WRITE (LINE, FMT75, PGNUM ) ;                                 00150600
        WRITE (LINE, 16 , TITLEARRAY[*] ) ;                           00150700
        WRITE (LINE, FMT90 ) ;                                        00150800
        WRITE (LINE, FMT12, TIME],NITER,  FOR I := 1 STEP 1           00150900
                     UNTIL NPRES DO [ ANGLE[I], PRES[2*I-1] ] ) ;     00151000
        IF EVEN = 1 THEN WRITE(LINE, FMT72, ANGLE[NPRES + 1] ,        00151100
                     PRES[2*NPRES] ) ;                                00151200
        WRITE (LINE, FMT13, PAREA, SAREA, CD );                       00151300
        NPP := 0 ;                                                    00151400
        END ;                                                         00151500
%*************************************************************************00151600
%                                                                     00151700
   PROCEDURE PRINTPARAMETERS ;                                        00151800
        BEGIN                                                         00151900
        PGNUM := PGNUM + 1 ;                                          00152000
        WRITE (LINE, FMT75, PGNUM ) ;                                 00152100
        WRITE (LINE, 16 , TITLEARRAY[*] ) ;                           00152200
        WRITE (LINE, FMT16 )  ;                                       00152300
        WRITE (LINE, FMT4) ;                                          00152400
                     % REYN::= REYNOLDS NUMBER                        00152500
                     % YINF::= LARGEST VALUE OF TS1,THE RADIAL CO-    00152600
```

```
                        %       ORDINATE                                        00152700
                        % LNP::= THE NUMBER OF TIME STEPS BETWEEN PRINT         00152800
                        %       OUTS MINUS ONE.                                 00152900
                        % NITER::= TIME STEP COUNTER, INITIALLY ZERO             00153000
                        % LNITER::= TOTAL NUMBER OF TIME STEPS FOR THIS         00153100
                        %       RUN                                             00153200
                        % TIME2::= MAXIMUM PROCESSOR TIME IN SECONDS             00153300
                        % DTIME ::= DIMENSIONLESS TIME INCREMENT                 00153400
                                                                                00153500
                        % NXEND ::= NUMBER OF MESH POINTS IN THE                 00153600
                        %       CIRCUMFERENTIAL  DIRECTION                      00153700
                        % NYEND ::= NUMBER OF MESH POINTS IN THE                 00153800
                        %       RADIAL DIRECTION                                00153900
                        % WFP ::= RELAXATION PARAMETER                           00154000
                        % TIME1 ::= INITIAL DIMENSIONLESS TIME                   00154100
                        % POTENTIAL ::= IF 1, OUTER BC TAKEN AS                  00154200
                        %       POTENTIAL FLOW, ELSE PARALLEL FLOW              00154300
        WRITE (LINE, FMT5, REYN, YINF, LNP, LNPP, LNITER, TIME2,                00154400
                           DTIME, TIME1, NITER,                                 00154500
                           POTENTIAL, NXEND, NYEND,IQX,                         00154600
                           IQY,DELX, DELY,STARTTIME,SAVE,                       00154700
                           DSV,PPZ,IREC,                                        00154800
                           SECTION ) ;                                          00154900
        END ;                                                                   00155000
%***************************************************************************00155100
%                                                                               00155200
   PROCEDURE OUTPUTRESULTS ;                                                    00155300
        BEGIN                                                                   00155400
        IF PZ=1 THEN PRINT1 ( ZETA, ZPRNT ) ;                                   00155500
        IF PP=1 THEN PRINT1 ( PSI, PPRNT ) ;                                    00155600
        IF DSV EQL 1 THEN SAVEPSIZETA ;                                         00155700
        NP := 0 ;                                                               00155800
        END ;                                                                   00155900
%***************************************************************************00156000
%                                                                               00156100
   PROCEDURE FINALPRINTROUTINE ;                                                00156200
        BEGIN                                                                   00156300
        IF NP NEQ 0 THEN BEGIN OUTPUTRESULTS; PRESS END ;                       00156400
        IF SAVE=1 THEN BEGIN;STOREPSIZETA;STORECONSTANTS;END;                   00156500
        LOCK(SAVEP) ;                                                           00156600
        LOCK(SAVEZ) ;                                                           00156700
        END ;                                                                   00156800
%***************************************************************************00156900
%                                                                               00157000
   PROCEDURE PRINTPSIZETA ;                                                     00157100
        BEGIN                                                                   00157200
        LABEL HOP165, HOP190 ;                                                  00157300
        SETUPGET ;                                                              00157400
        GETPSIZETA ;                                                            00157500
        OUTPUTRESULTS ;                                                         00157600
        PRESS ;                                                                 00157700
 HOP165:NITER := NITER + 1 ;                                                    00157800
        NP := NP + 1 ;                                                          00157900
        NPP := NPP + 1 ;                                                        00158000
        TIME1 := TIME1 + DTIME ;                                                00158100
        GETPSIZETA ;                                                            00158200
              % PRINT CURRENT VORTICITY AND STREAMFUNCTION VALUES                00158300
              %    AND COMPUTE THE PRESSURE DISTRIBUTION                        00158400
                        %    WHENEVER REQUESTED BY LNP                          00158500
        IF NP EQL LNP THEN OUTPUTRESULTS;                                       00158600
```

```
        IF NPP EQL LNPP THEN PRESS ;                                    00158700
                        % IF PROGRAM RUNS OVERTIME GO TO END OF JOB      00158800
        IF  GOP GEQ 1.5  THEN  GO TO HOP190 ;                           00158900
                        % IF DESIRED NUMBER OF TIME STEPS HAVE          00159000
                        %    BEEN COMPLETED THEN GO TO END OF JOB       00159100
        IF NITER GEQ LNITER  THEN  GO TO HOP190 ;                       00159200
                                                                        00159300
                        % OTHERWISE COMPUTE THE NEXT TIME STEP          00159400
        GO TO HOP165 ;                                                  00159500
 HOP190:END ;                                                           00159600
%***********************************************************************00159700
%                                                                       00159800
   PROCEDURE MAIN ;                                                     00159900
        BEGIN                                                           00160000
        LABEL  HOP165, HOP190 ;                                         00160100
        INITIALIZEPROGRAM ;                                             00160200
        COMPUTECONSTANTS ;                                              00160300
        PRINTPARAMETERS ;                                               00160400
        IF PPZ EQL 1 THEN BEGIN PRINTPSIZETA;GO TO HOP190;END;          00160500
        INITIALIZENET ;                                                 00160600
        IF SECTION NEQ 1  THEN GO TO HOP165 ;                           00160700
        OUTPUTRESULTS ;                                                 00160800
        PRESS ;                                                         00160900
                        % MAIN PROGRAM LOOP                             00161000
 HOP165:NITER := NITER + 1 ;                                            00161100
        NP := NP + 1 ;                                                  00161200
        NPP := NPP + 1 ;                                                00161300
        TIME1 := TIME1 + DTIME ;                                        00161400
                        % COMPUTE THE NEW VORTICITY BY THE              00161500
                        %    ALTERNATING DIRECTION IMPLICIT METHOD      00161600
        VORTE ;                                                         00161700
        VORTS ;                                                         00161800
                        % RELAX THE STREAM FUNCTION WITH THE            00161900
                        %    ACCELERATED LIEBMAN METHOD                 00162000
        STREAM ;                                                        00162100
                        % COMPUTE THE SURFACE VORTICITY                 00162200
        SURFACEVORTICITY ;                                              00162300
              % PRINT CURRENT VORTICITY AND STREAMFUNCTION VALUES        00162400
              %    AND COMPUTE THE PRESSURE DISTRIBUTION               00162500
                        %    WHENEVER REQUESTED BY LNP                  00162600
        IF NP EQL LNP THEN OUTPUTRESULTS ;                              00162700
        IF NPP EQL LNPP THEN PRESS ;                                    00162800
                        % IF PROGRAM RUNS OVERTIME GO TO END OF JOB      00162900
        CHECKTIMELIMIT ;                                                00163000
                        % IF DESIRED NUMBER OF TIME STEPS HAVE          00163100
                        %    BEEN COMPLETED THEN GO TO END OF JOB       00163200
        IF NITER GEQ LNITER  THEN  GO TO HOP190 ;                       00163300
                                                                        00163400
                        % OTHERWISE COMPUTE THE NEXT TIME STEP          00163500
        GO TO HOP165 ;                                                  00163600
 HOP190:END ;                                                           00163700
%***********************************************************************00163800
%***********************************************************************00163900
%                                                                       00164000
        MAIN ;                                                          00164100
 HOP999:FINALPRINTROUTINE ;                                             00164200
 HOPEND:TYMOUT ;                                                        00164300
        END ;                                                           00164400
%***********************************************************************00164500
%                                                                       00164600
```

```
END.                                                    00164700
```

# APPENDIX E

## CICHY/UNS

```
   BEGIN
* DECLARATION OF GLOBAL VARIABLES
                        PE REAL SUBROUTINE
          ENTERSYSTEMPARAMETERS :
               BEGIN
               NREY := 40.0   .;
               TLMT := 0.08   ;
               LIMIT := 0.00001 ;
               NR1 :=      ;
               NX2 :=      ;
               NX3 :=      ;
               NX4 :=      ;
               NX5 :=      ;
               NE1 :=      ;
               NY2 :=      ;
               NY3 :=      ;
               NY4 :=      ;
               NY5 :=      ;
               PINC := 1   ;
               END ;
                        PE REAL SUBROUTINE
          INITIALIZECOORDINATES ;
               BEGIN
               FILL R1 WITH
                          2.0,       1.9,       1.8,       1.7,
                          1.6,       1.5,       1.4,       1.3,
                          1.22,      1.160,     1.11637,   1.0774,
                          1.0515,    1.03424,   1.0226,    1.01497,
                          1.00985,   1.00643,   1.00416,   1.00264,
                          1.001625,  1.00095,   1.0005,    1.0002,
                          1.0   ;
               FILL X2 WITH
                         -11.0,     -10.0,     -9.0,      -8.0,
                         -7.0,      -6.0,      -5.0,      -4.245,
                         -3.692,    -3.288,    -2.992,    -2.775,
                         -2.616,    -2.5,      -2.4,      -2.3,
                         -2.2,      -2.1,      -2.0,      -1.97538,
                         -1.90211,  -1.78201,  -1.61803,  -1.41421,
                         -1.17557,  -0.90798,  -0.61803,  -0.31287,
                          0.0,      +0.31287   ;
               FILL X3 WITH
                         -0.31287,   0.0,      +0.31287,   0.44441,
                          0.62450,   0.75993,   0.87178,   0.96825,
                          1.05357,   1.13027,   1.200,     1.26392,
                          1.32288,   1.37750,   1.42829,   1.47564,
                          1.51987,   1.56125,   1.60,      1.63631,
                          1.67033,   1.70220,   1.73205,   1.75997,
                          1.78606,   1.81039,   1.83303,   1.85405,
                          1.87350,   1.89143,   1.90788,   1.92289,
                          1.93649,   1.94872,   1.95960,   1.96914,
                          1.97737,   1.98431,   1.98997,   1.99437,
                          1.99750,   1.99937,   2.0,       2.00017,
                          2.000425,  2.000807,  2.00138,   2.00224,
                          2.003533,  2.005469,  2.00837,   2.01273,
                          2.01926,   2.02907,   2.04377,   2.06583,
                          2.0989,    2.14854,   2.2229,    2.3   ;
               LOOP J:=60,1,136  DO  X3[J] := X3[J-1] + 0.1 ;
%                        X3(136) IS AT X=10, X3(60) IS AT X=2.4
               X4[0] := X3[135] ;
%                       X4(0) IS AT X=9.9
```

```
LOOP J:=1,1,151  DO  X4[J] := X4[J-1] + 0.1 ;
%           X4(151) IS AT X=25
%
X5[0] := X4[149] ;
%           X5(0) IS AT X=24.8
LOOP J:=1,1,126  DO  X5[J] := X5[J-1] + 0.2 ;
%           X5(126) IS AT X=50
%
%           REGION 1 HAS  25 MESH ROWS, R1=  2 TO R1=  1
%           REGION 2 HAS  29 MESH ROWS, X2=-11 TO X2=  0
%           REGION 3 HAS 136 MESH ROWS, X3=  0 TO X3= 10
%           REGION 4 HAS 151 MESH ROWS, X4= 10 TO X4= 25
%           REGION 5 HAS 126 MESH ROWS, X5= 25 TO X5= 50
%
%
FILL E1[0] WITH
            0.0,       0.0,       0.0,       0.0,       0.0,
           +1.95,      0.0,       0.05,      0.1,       0.15,
            0.2,       0.25,      0.3,       0.35,      0.4,
            0.45,      0.5,       0.55,      0.57133,   0.60108,
            0.62406,   0.64357,   0.66086,   0.67660,   0.69118,
            0.70483,   0.71775,   0.73005,   0.74184,   0.75318,
            0.76414,   0.77477,   0.78510,   0.79517,   0.80500,
            0.81463,   0.82407,   0.83333,   0.84245,   0.85142,
            0.86027,   0.86901,   0.87764,   0.88618,   0.89464,
            0.90301,   0.91132,   0.91957,   0.92776,   0.93591,
            0.94401,   0.95207,   0.96011,   0.96812,   0.97610,
            0.98408,   0.99204,   1.0 ;
FILL E1[1] WITH
            0.0,       0.0,       0.0,       0.0,       0.0,
            1.0,       1.00796,   1.01592,
            1.02390,   1.03188,   1.03989,   1.04793,   1.05600,
            1.06409,   1.07224,   1.08043,   1.08868,   1.09699,
            1.10536,   1.11381,   1.12236,   1.13099,   1.13973,
            1.14858,   1.15755,   1.16667,   1.17593,   1.18537,
            1.19500,   1.20483,   1.21490,   1.22523,   1.23586,
            1.24682,   1.25816,   1.26995,   1.28225,   1.29517,
            1.30882,   1.32340,   1.33914,   1.35643,   1.37594,
            1.39892,   1.42867,   1.450,     1.50,      1.55,
            1.6,       1.65,      1.7,       1.75,      1.8,
            1.85,      1.9,       1.95,      0.0,       0.05  ;
FILL Y2[0] WITH
           +14.0,      13.0,      12.0,      11.0,
            10.0,       9.0,       8.0,       7.0,
             6.0,       5.0,       4.245,     3.692,
             3.288,     2.992,     2.775,     2.616,
             2.5,       2.4,       2.3,       2.2,
             2.1,       2.0,       1.97538,   1.90211,
             1.78201,   1.61803,   1.41421,   1.17557,
             0.90798,   0.61803,   0.31287,   0.0,
            -0.31287,  -0.61803,  -0.90798,  -1.17557,
            -1.41421,  -1.61803,  -1.78201,  -1.90211,
            -1.97538,  -2.0,      -2.1,      -2.2,
            -2.3,      -2.4,      -2.5,      -2.616,
            -2.775,    -2.992,    -3.288,    -3.692,
            -4.245,    -5.0,      -6.0,      -7.0,
            -8.0,      -9.0,     -10.0,     -11.0,
           -12.0,     -13.0,     -14.0  ;
FILL Y3[0] WITH
            0.0,       0.0,       0.0,       0.0,       0.0,
```

```
                +14.0,      13.0,      12.0,      11.0,
                 10.0,      9.0,       8.0,       7.0,       6.0,
                 5.0,       4.6225,    4.245,     3.9685,    3.692,
                 3.490,     3.288,     3.140,     2.992,     2.8835,
                 2.775,     2.6955,    2.616,     2.558      2.5,
                 2.45,      2.4,       2.35,      2.3,       2.25,
                 2.2,       2.15,      2.1,       2.05,      2.0,
                 1.97538,   1.95,      1.90,      1.85,      1.80,
                 1.75,      1.70,      1.65,      1.60,      1.55,
                 1.50,      1.45,      1.40,      1.35,      1.30,
                 1.25 ;
FILL Y3[1] WITH
                 0.0,       0.0,       0.0,       0.0,       0.0,
                 1.20,      1.15,      1.10,      1.05,
                 1.00,      0.95,      0.90,      0.85,      0.80,
                 0.75,      0.70,      0.65,      0.60,      0.55,
                 0.50,      0.45,      0.40,      0.35,      0.30,
                 0.25,      0.20,      0.15,      0.10,      0.05,
                 0.0,      -0.05,     -0.10,     -0.15,     -0.20,
                -0.25,     -0.30,     -0.35,     -0.40,     -0.45,
                -0.50,     -0.55,     -0.60,     -0.65,     -0.70,
                -0.75,     -0.80,     -0.85,     -0.90,     -0.95,
                -1.00,     -1.05,     -1.10,     -1.15,     -1.20,
                -1.25 ;
FILL Y3[2] WITH
                 0.0,       0.0,       0.0,       0.0,       0.0,
                -1.30,     -1.35,     -1.40,     -1.45,
                -1.50,     -1.55,     -1.60,     -1.65,     -1.70  ;
                -1.75,     -1.80,     -1.85,     -1.90,
                -1.95,     -1.97538,  -2.00,     -2.05,
                -2.10,     -2.15,     -2.20,     -2.25,     -2.30,
                -2.35,     -2.40,     -2.45,     -2.50,     -2.558,
                -2.616,    -2.6955,   -2.775,    -2.8835,   -2.992,
                -3.140,    -3.288,    -3.490,    -3.692,    -3.9685,
                -4.245,    -4.6225,   -5.0,      -6.0,      -7.0,
                -8.0,      -9.0,      -10.0,     -11.0,     -12.0,
                -13.0,     -14.0  ;
FILL Y4[0] WITH
                 0.0,       0.0,       0.0,       0.0,       0.0,
                +14.0,      13.0,      12.0,      11.0,
                 10.0,      9.0,       8.0,       7.0,       6.0,
                 5.0,       4.6225,    4.245,     3.9685,    3.692,
                 3.490,     3.288,     3.140,     3.00,      2.90,
                 2.8,       2.7,       2.6,       2.5,       2.4,
                 2.3,       2.2,       2.1,       2.0,       1.9,
                 1.8,       1.7,       1.6,       1.5,       1.4,
                 1.3,       1.2,       1.1,       1.0,       0.9,
                 0.8,       0.7,       0.6,       0.5,       0.4,
                 0.3,       0.2,       0.1,       0.0  ;
FILL Y4[1] WITH
                 0.0,       0.0,       0.0,       0.0,       0.0,
                -0.1,
                -0.2,      -0.3,      -0.4,      -0.5,      -0.6
                -0.7,      -0.8,      -0.9,      -1.0,      -1.1,
                -1.2,      -1.3,      -1.4,      -1.5,      -1.6,
                -1.7,      -1.8,      -1.9,      -2.0,      -2.1,
                -2.2,      -2.3,      -2.4,      -2.5,      -2.6,
                -2.7,      -2.8,      -2.9,      -3.0       -3.140,
                -3.288,    -3.490,    -3.692,    -3.9685,   -4.245,
                -4.6225,   -5.0,      -6.0,      -7.0,      -8.0,
```

```
                -9.0,      -10.0,     -11.0,     -12.0,    -13.0,
                -14.0  ;
     FILL Y5[0] WITH
                +14.0,      13.0,      12.0,      11.0,
                 10.0,      9.0,       8.0,       7.0,      6.0,
                 5.0,       4.245,     3.9685,    3.692,    3.490,
                 3.288,     3.0,       2.8,       2.6,      2.4,
                 2.2,       2.0,       1.8,       1.6,      1.4,
                 1.2,       1.0,       0.8,       0.6,      0.4,
                 0.2,       0.0,      -0.2,      -0.4,     -0.6,
                -0.8,      -1.0,      -1.2,      -1.4,     -1.6,
                -1.8,      -2.0,      -2.2,      -2.4,     -2.6,
                -2.8,      -3.0,      -3.288,    -3.490,   -3.692,
                -3.9685,   -4.245,    -5.0,      -6.0,     -7.0,
                -8.0,      -9.0,      -10.0,     -11.0,    -12.0,
                -13.0,     -14.0  ;
                PE REAL SUBROUTINE
COMPUTECONSTANTS ;
     BEGIN
     DEFINE AR  &A1 &IA:=1.0/(&A1[&IA] * (RTR(1,,&A1[&IA])
                + &A1[&IA])) ## ;
     DEFINE AL  &A1 &IA:=1.0/(&A1[&IA] * (RTL(1,,&A1[&IA])
                + &A1[&IA])) ## ;
     DEFINE BR  &II &LIM &BM &B1:=
          BEGIN
          LOOP  IL:=&II,1,&LIM  DO
                BEGIN
                &BM[IL]  := 1.0/(&B1[IL] * (&B1[IL-1]+&B1[IL]));
                END ## ;
     DEFINE BL  &II &LIM &BP &B1:=
          BEGIN
          LOOP IL:=&II,1,&LIM DO
                BEGIN
                &BP[IL]  := 1.0/(&B1[IL] * (&B1[IL+1]+&B1[IL]));
                END ## ;
                THE CELL HEIGHT A[S] AND CELL WIDTH B[J]
                MUST BE DETERMINED FROM AN IMPLICIT RELATIONSHIP
                AMONG THE COORDINATES IN A ROW OR COLUMN  SO
                THAT THE MESH POINT IS CENTERED IN THE CELL.
                IT IS ASSUMED HERE THAT THIS IMPLICITE RELATION
                IS SOLVED ELSEWHERE AND THAT THE APPROPRIATE
                VALUES OF A AND B ARE TO BE ENTERED WITH
                FILL STATEMENTS
     FILL B1 WITH
     FILL B2 WITH
     FILL B3 WITH
     FILL B4 WITH
     FILL B5 WITH
     FILL A1[0] WITH
     FILL A1[1] WITH
     FILL A2[0] WITH
     FILL A3[0] WITH
     FILL A3[1] WITH
     FILL A3[2] WITH
     FILL A4[0] WITH
     FILL A4[1] WITH
     FILL A5[0] WITH
     AM1[0] := AR A1 0 ;
     AM1[1] := AR A1 1 ;
     AM2[0] := AR A2 0 ;
```

```
          AM3[0] := AR A3 0 ;
          AM3[1] := AR A3 1 ;
          AM3[2] := AR A3 2 ;
          AM4[0] := AR A4 0 ;
          AM4[1] := AR A4 1 ;
          AM5[0] := AR A5 0;
          AP1[0] := AL A1 0 ;
          AP1[1] := AL A1 1 ;
          AP2[0] := AL A2 0 ;
          AP3[0] := AL A3 0 ;
          AP3[1] := AL A3 1 ;
          AP3[2] := AL A3 2 ;
          AP4[0] := AL A4 0 ;
          AP4[1] := AL A4 1 ;
          AP5[0] := AL A5 0 ;
          BR 0 NR1 BM1 B1  ;
          BR 0 NX2 BM2 B2  ;
          BR 0 NX3 BM3 B3  ;
          BR 0 NX4 BM4 B4  ;
          BR 0 NX5 BM5 B5  ;
          BL 0 NR1 BP1 B1 ;
          BL 0 NX2 BP2 B2 ;
          BL 0 NX3 BP3 B3 ;
          BL 0 NX4 BP4 B4 ;
          BL 0 NX5 BP5 B5 ;
%                 SET UP MODE CONTROL CONSTANTS FOR EACH ROW
%                 IN REGION 2. EXCLUDE ALL POINTS INCIDE CIRCLE
          LOOP I:=0,1,18 DO
               M2[I] := BOOLEAN ( 7FFFFFFFFFFFFFFD(16)) ;
          CB := BOOLEAN (0000000100000000(16)) ;
          NCB := CB ;
%                 GENERATE THE MODE PATTERNS FOR THE PART
%                 OF THE REGION WHERE THE CYLINDRICAL
%                 COORDINATES ARE
          LOOP I:=19,1,28 DO
               BEGIN
               M2[I] := M2[18] AND NOT NCB ;
               NCB := (SHIFTL(1,NCB) ) OR CB ;
               CB := SHIFTR(1,CB) ;
               NCB := NCB AND CB ;
               END
          M2[29] := M2[27] ;
          BODD1 := BOOLEAN(5555555555555555(16)) ;
          BEVEN1 := BOOLEAN(0AAAAAAAAAAAAAAA(16));
          M2BC     := BOOLEAN(8000000000000002(16)) ;
          M2BCL     := BOOLEAN(8000000000000000(16)) ;
          M2BCR     := BOOLEAN(0000000000000002(16)) ;
          END ;
                 PE REAL SUBROUTINE
     STREAM ;
          BEGIN
                 PE REAL SUBROUTINE
          NEWPSI (PCPOINT PSI, PCPOINT ZETA, PINT J, PINT K,
                 PCPOINT BM, PCPOINT BP, PCPOINT AM, PCPOINT AP
                 PINT IY, PINT IX, CINT OUT LIM, BOOLEAN PMODE,
                 CINT S ) ;
               BEGIN
               MODE := PMODE ;
               TEST := PSI[J] ;
               ID := 1.0 / (BM[IX] + BP[IX] + AM[IY] + AP[IY] ) ;
```

```
                PSI[J] := ID * (0.5 * ZETA[J] + BM[IX] *
                     RTR(1,,PSI[K]) + BP[IX] * RTL(1,,PSI[K])
                     + AM[IY] * PSI[J-S] + AP[IY] * PSI[J+S] ) ;
                DIF := PSI[J] - TEST ;
                IF ABS(DIF) GTR LIMIT THEN LIM := 1 ;
                END ;
%                    THE ITERATIVE SOLUTION OF THE STREAM FUNCTION
%                    EQUATION CYCLES THROUGH THE 4 SECTIONS
                     PE REAL SUBROUTINE
           STREAM1 ;
                BEGIN
%                    TREAT THE INFLOW BOUNDARY
                MODE := M2[0] ;
                PSA[0] := PSA[1] ;
%                    SET UP THE TOP AND BOTTOM BOUNDARIES
                MODE := M2BCL ;
                LOOP J:=1,1,28  DO
                     BEGIN
                     PSA[J] := RTL(1,,PSA[J]) + 1.0 ;
                     END ;
                MODE := M2BCR ;
                LOOP J:=1,1,28  DO
                     BEGIN
                     PSA[J] := RTR(1,,PSA[J]) + 1.0 ;
                     END ;
%
%                    SWEEP THE ODD POINTS IN THE MESH
                SW1 := CEVEN2 ;
                SW2 := CODD2
                PASS := 1 ;
                A := 1 ;
PASS2:          S := 1 ;  % NUMBER OF PE ROWS PER MESH ROW
                IY := 0 ;
                LOOP C:=1,2,28  DO
                     BEGIN
                     J := C + SW1 ; * ROWS   BEING UPDATED
                     K := C + SW2 ; % ADJACENT POINTS
                                    % BOTH RELATIVE TO BUFFER ORIGIN
                     PMODE := M2[C+A] ;
                     IX := J ;
                     NEWPSI (PSA, ZSA, J, K, BM2, BP2, AM2, AP2,
                                    IY, IX, LIM, PMODE, S ) ;
                     END ;
                IF PASS=1 THEN
                     BEGIN
                     PASS := 2 ;
                     A := 0 ;
                     SW1 := CODD2 ;
                     SW2 := CEVEN2 ;
                     GO TO PASS2 ;
                     END ;
%
%                    NOW MOVE PSA[28] TO A WORK AREA, EXPAND IT
%                    BY INTERPOLATION SO IT CORRESPONDS TO THE MESH
%                    ROW WIDTH IN REGION 3, AND INSERT IT IN PSA[38]
                END ;
                     PE REAL SUBROUTINE
           STREAM2 ;
                BEGIN
                END ;
```

```
                    PE REAL SUBROUTINE
               STREAM3 ;
                    BEGIN
                    END ;
                    PE REAL SUBROUTINE
               STREAM4 ;
                    BEGIN
                    END ;
               NITER := 0 ;
ITERATE:            BEGIN
                    LIM := 0 ;
                    STREAM1 ;
                    STREAM2 ;
                    STREAM3 ;
                    STREAM4 ;
                    IF LIM = 0 THEN GO TO CONVERGED ;
                    NITER := NITER + 1 ;
                    GO TO ITERATE ;
CONVERGED:          END ;
                    PE REAL SUBROUTINE
          SURFACEVORTICITY ;
               BEGIN
               END ;
                    PE REAL SUBROUTINE
          INITIALIZESTREAM ;
               BEGIN
               DEFINE INS &X2 &Y2 &I2 &J2:=
                    &Y2[&J2] * (1.0 - 1.0 / (&X2[&I2]*&X2[&I2]
                    +&Y2[&J2]*&Y2[&J2])) ##
                    PE REAL SUBROUTINE
               INITIAL1 ;
                    BEGIN
                    LOOP K:=1,1,29  DO
                         BEGIN
                         MODE := M2[K] ;
                         PSA[K] := INS X2 Y2 K 0 ;
                         ZSA[K] := 0.0 ;
                         END ;
                    L := 0     % INDICATES ROW IN PE IN A REGION
                    N := 0     % INDICATES ROW IN REGION
                    LOOP K:=35,3,247  DO       % UP TO X= 3.4
                         BEGIN
                         LOOP W:=0,1,2 DO
                              BEGIN
                              MODE := M3[L] ;
                              PSA[K+W] := INS X3 Y3 N W  ;
                              ZSA[K+W] := 0.0 ;
                              L := L + 1 ;
                              END ;
                         N := N + 1 ;
                         END ;
                    LOOP I:=0,1,2  DO
                         BEGIN
                         PPS1[I] := INS X3 Y3 70 I ;
                         PZS1[I] := 0.0 ;
%
%                        WRITE OUT 512 ROWS (32 PAGES)
%                        STARTING WITH PSA(0) THROUGH
%                        PSA[255] AND FROM ZSA[0] THROUGH
%                        ZSA(255)
```

```
%
%                    SINCE I/O IS NOT CLEARLY SPECIFIED
%                    AT THIS TIME. THIS SECTION MUST BE CODED
%                    LATER. IN ORDER TO HAVE RAPID EXCHANGES
%                    OF DATA TO AND FROM DISK, AND TRANSFERS AND
%                    CALCULATIONS MUST OVERLAP.
%                    AFTER SECTION 1 HAS BEEN WRITTEN ON DISK
%                    AND WHILE COMPUTATIONS ARE STILL TAKING
%                    PLACE ON SECTION 2 IN BUFFER B, READ
%                    SECTION 3 INTO BUFFER A
%
               END ;
%
                    PE REAL SUBROUTINE
          INITIAL2 ;
               BEGIN
               N := N - 2 ;
               LOOP K:=0,3,203 DO     % STARTS X=3.3 ENDS X=10
                    BEGIN
                    LOOP W:=0,1,2  DO
                         BEGIN
                         MODE := M3C[W] ;
                         PSB[K+W] := INS X3 Y3 N W ;
                         ZSB[K+W] := 0.0 ;
                         END ;
                    N := N + 1 ;
                    END ;
               N : = 0 ;
               LOOP  K:= 215,2,254  DO  % START X=9.9 END X=11.8
                    BEGIN
                    LOOP W:=0,1,1  DO
                         BEGIN
                         MODE := M4[W] ;
                         PSB[K+W] := INS X4 Y4 N W :
                         ZSB[K+W] := 0.0 ;
                         END ;
                    N := N + 1 ;
                    END ;
%                   WRITE SECTION 2 ON DISK AND
%                   READ SECTION 4 IN TO BUFFER B
               END ;
                    PE REAL SUBROUTINE
          INITIAL3 ;
               BEGIN
               N := N-2 ;
               LOOP K:=0,2,243  DO  % START X=11.7  END X=23.8
                    BEGIN
                    LOOP W:=0,1,1  DO
                         BEGIN
                         MODE := M4[W] ;
                         PSA [K+W] := INS X4 Y4 N W ;
                         ZSA [K+W] := 0.0 :
                         END ;
                    N := N + 1 ;
                    END ;
%                   WRITE SECTION 3 ON DISK AND
%                   READ SECTION 1 INTO BUFFER A
               END ;
                    PE REAL SUBROUTINE
          INITIAL4 ;
```

```
                BEGIN
                N := N - 2 ;
                LOOP K:=0,2,27  DO  % START X=23.7 END X=25
                     BEGIN
                     LOOP W:=0,1,1  DO
                          BEGIN
                          MODE := M4[W] ;
                          PSB[K+W] := INS X4 Y4 N W ;
                          ZSB[K+W] := 0.0 ;
                          END ;
                     N := N + 1 ;
                     END ;
                N := 0 ;
                LOOP K:=30,1,155  DO   % START X=24.8 ENDS X=49.8
                     BEGIN
                     MODE := M5 ;
                     PSB[K] := INS X5 Y5 N 0 ;
                     ZSB[K] := 0.0 ;
                     N := N + 1 ;
                     END ;
                N := 0 ;
                LOOP K:=160,2,209  DO  % START R=2 ENDS R=1
                     BEGIN
                     LOOP W:=0,1,1  DO
                          BEGIN
                          MODE := M1[W] ;
                          PSB[K+W] := R1[N] * SIN(E1[W])
                                      *(1.0-1.0/(R1[N]*R1[N])) ;
                          ZSB[K+W] := 0.0 ;
                          END ;
                     N := N + 1 ;
                     END ;
%                    ASSIGN THE SURFACE STREAM FUNCTION THE
%                    THE VALUE ZERO
                MODE := M1[0] ;
                PSB[208] := 0.0 ;
                MODE := M1[1] ;
                PSB[209] := 0.0 ;
%                    WRITE SECTION 4 ON DISK AND
%                    READ SECTION 2 INTO BUFFER B
                END ;
           INITIAL1 ;
           INITIAL2 ;
           INITIAL3 ;
           INITIAL4 ;
           END ;
                    PE REAL SUBROUTINE
      DELTAT ;
           BEGIN
           END ;
                    PE REAL SUBROUTINE
      VORTICITY ;
           BEGIN
                    PE REAL SUBROUTINE
           VORT ;
                BEGIN
                ZETA[J] :=-2.0 *((RTL(1,,PSI[J]) - PSI[J]) / BP[J]
                     - (PSI[J] - RTR(1,,PSI[J])) / BM[J]
                     + (PSI[J+1] - PSI[J]) / AP[J]
                     - (PSI[J] - PSI[J-1]) / AM[J] )  ;
```

```
                END ;
                     PE REAL SUBROUTINE
           VORT1 ;
                BEGIN
                END ;
                     PE REAL SUBROUTINE
           VORT2 ;
                BEGIN
                END ;
                     PE REAL SUBROUTINE
           VORT3 ;
                BEGIN
                END ;
                     PE REAL SUBROUTINE
           VORT4 ;
                BEGIN
                END ;
           END ;
                     PE REAL SUBROUTINE
      PRINTOUT ;
           BEGIN
           END ;
                     PE REAL SUBROUTINE
      PROGRAMCONTROL ;
           BEGIN
           END ;
                     PE REAL SUBROUTINE
 MAIN ;
      BEGIN
      INITIALIZECONSTANTS ;
      INITIALIZECOORDINATES ;
      INITIALIZESTREAM ;
      SURFACEVORTICITY ;
NXTSTP:
      DELTAT ;
      VORTICITY ;
      STREAM ;
      SURFACEVORTICITY ;
      PRINTOUT ;
      PROGRAMCONTROL ;
      END ;
 MAIN ;
 END.
```

UNCLASSIFIED
Security Classification

**DOCUMENT CONTROL DATA - R & D**

*(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)*

| 1. ORIGINATING ACTIVITY (Corporate author) | 2a. REPORT SECURITY CLASSIFICATION |
|---|---|
| Center for Advanced Computation<br>University of Illinois at Urbana-Champaign<br>Urbana, Illinois 61801 | UNCLASSIFIED |
| | 2b. GROUP |

3. REPORT TITLE

The Use of ILLIAC IV to Solve Numerical Fluid Dynamics Problems

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Research Report

5. AUTHOR(S) (First name, middle initial, last name)

Paul T. Cichy

| 6. REPORT DATE | 7a. TOTAL NO. OF PAGES | 7b. NO. OF REFS |
|---|---|---|
| February 27, 1973 | 149 | 5 |

| 8a. CONTRACT OR GRANT NO. | 9a. ORIGINATOR'S REPORT NUMBER(S) |
|---|---|
| DAHC04-72-C-0001<br>b. PROJECT NO.<br>ARPA Order No. 1899<br>c.<br>d. | CAC Document No. 67<br>9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) |

10. DISTRIBUTION STATEMENT

Copies of this report may be requested from the address given in (1) above.

| 11. SUPPLEMENTARY NOTES | 12. SPONSORING MILITARY ACTIVITY |
|---|---|
| | U. S. Army Research Office - Durham<br>Duke Station, Durham, North Carolina |

13. ABSTRACT

The usefulness of ILLIAC IV in solving numerical fluid dynamics problems is studied by developing a parallel algorithm for the unsteady two dimensional flow of an incompressible fluid around a circular cylinder. Two cases are studied, the symmetric case, for which an operational GLYPNIR program is included, and the assymetric case, for which a partially completed GLYPNIR program is attached. The programming of a parallel algorithm for ILLIAC IV is more difficult than programming a comparable serial algorithm but this additional difficulty is compensated for by the greatly reduced computation time required to solve the problem.

DD FORM 1 NOV 68 1473

UNCLASSIFIED
Security Classification

| 14. KEY WORDS | LINK A | | LINK B | | LINK C | |
|---|---|---|---|---|---|---|
| | ROLE | WT | ROLE | WT | ROLE | WT |
| ILLIAC IV | | | | | | |
| GLYPNIR | | | | | | |
| Fluid Dynamics | | | | | | |
| Parallel Algorithm | | | | | | |

BOUND BY
HNM

UNIVERSITY OF ILLINOIS-URBANA
510.84IL63C C001
CAC DOCUMENT$UR8ANA
61-70 1973
3 0112 007264150